数据仓库与数据挖掘

原理及应用（第二版）

郑 岩 编著

DATA WAREHOUSE AND DATA MINING

(Second Edition)

清华大学出版社

数据仓库与数据挖掘原理及应用（第二版）

郑　岩　编著

清华大学出版社
北　京

内 容 简 介

本书全面地介绍数据仓库和数据挖掘的原理及其应用，系统地阐述数据仓库和数据挖掘的主要概念和算法等基础知识，并结合当前各领域的具体应用实例进一步帮助广大读者加深理解，力求学以致用。

全书分为3篇。第1篇介绍数据仓库的发展和演变，主要阐述数据仓库的定义、体系结构、组成、数据模型和ETL过程等，描述数据仓库的设计方法和实现过程，结合实例说明如何构建数据仓库，扼要地介绍数据仓库的应用，如OLAP和OLAM。第2篇介绍数据挖掘的起源和发展，主要阐述数据挖掘和Web挖掘的主要算法，包括聚类、分类、预测和关联分析等，描述如何运用数据挖掘解决实际问题，如客户细分、虚开欺诈识别和WAP日志挖掘等。第3篇阐述数据、信息和知识之间的关系，介绍当前研究热点——语义网和本体的核心技术及其主要应用。

本书可作为计算机及相关专业的研究生和高年级本科生教材，也可以作为计算机研究和开发人员以及相关专业人士的参考资料。

图书在版编目(CIP)数据

数据仓库与数据挖掘原理及应用/郑岩编著. —2版. —北京：清华大学出版社，2015
ISBN 978-7-302-37861-7

Ⅰ. ①数… Ⅱ. ①郑… Ⅲ. ①数据库系统 ②数据采集 Ⅳ. ①TP311.13 ②TP274

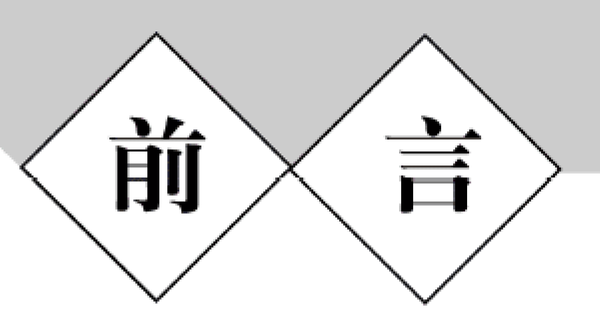
中国版本图书馆CIP数据核字(2014)第202718号

责任编辑：刘向威 李 晔
封面设计：
责任校对：梁 毅
责任印制：

出版发行：清华大学出版社
网 址：http://www.tup.com.cn，http://www.wqbook.com
地 址：北京清华大学学研大厦A座 **邮 编：**100084
社 总 机：010-62770175 **邮 购：**010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 刷 者：
装 订 者：
经 销：全国新华书店
开 本：185mm×260mm **印 张：**24 **字 数：**600千字
版 次：2015年1月第2版 **印 次：**2015年1月第1次印刷
印 数：1～ 000
定 价： .00元

产品编号：060519-01

0101010101010101010101010101010101
010101010101010101010101010101010101



前言

数据仓库是将海量数据进行抽取、清洗和转换，并按主题重新组织，可比喻成随时间推移不断丰富的“宝藏”；数据挖掘是从海量数据中发现人们感兴趣的知识，这些知识是隐含的、事先未知的潜在有用信息，并表示为概念、规则、规律和模式等，可比喻成“淘宝”。随着Internet的迅速普及和广泛应用，每天都会产生大量各种各样的数据，但它们背后到底隐藏着什么，这驱使人们不断探索其中的奥秘。

“工欲善其事，必先利其器”。在当今信息爆炸的时代，数据挖掘堪比“利器”，让我们在大数据时代不再感到茫然和困惑。未来，数据挖掘将绽放无限的生机和活力，可以辅助、部分代替甚至拓展人的智能和决策，以造福人类。

数据经整合汇总为信息，信息经挖掘抽象为知识，知识是智能的基石。因此，从信息化到知识化再到智能化将是人类社会发展的必然趋势。数据仓库和数据挖掘技术已经逐步渗透和深入到社会生活的各个方面，并不断催生出新的应用。

本书介绍数据仓库和数据挖掘的原理及其应用；此外，用较多篇幅列举数据仓库和数据挖掘在多个领域的实际应用。

全书分为3篇。第1篇介绍数据仓库的起源和演变，主要阐述数据仓库的定义、体系结构、组成、元数据、数据粒度、数据模型、数据质量及ETL过程，描述数据仓库设计方法和实现过程，结合实际详细阐述如何构建数据仓库及其主要应用，如OLAP和OLAM。第2篇介绍数据挖掘的起源和发展，主要阐述数据挖掘和Web挖掘的主要算法，包括聚类、分类、预测和关联分析等，详细描述数据挖掘的具体应用实例，如客户细分、虚开欺诈识别和WAP日志挖掘等。第3篇阐述数据、信息和知识之间的关系，介绍当前研究热点——语义网和本体的核心技术及方法。

本书作者潜心撰写，历时多年完成，旨在奉献精品以飨广大读者。由于水平有限，不当之处恳请赐教。

作　者

2014年8月

目录

第一篇　数据仓库

第 1 章　数据仓库基础　3

1.1　概述　3

1.1.1　演变　3

1.1.2　定义　5

1.2　体系结构　6

1.2.1　两层的体系结构　6

1.2.2　三层的体系结构　8

1.3　组成　9

1.3.1　加载管理器　10

1.3.2　仓库管理器　11

1.3.3　查询管理器　13

1.4　元数据　14

1.4.1　定义和分类　14

1.4.2　标准化　15

1.4.3　CWM　16

1.4.4　UML、MOF 和 XMI 与 CWM 的关系　20

1.5　数据粒度　22

1.6　数据模型　23

1.7　ETL 过程　23

1.7.1　主要流程　24

1.7.2　数据抽取　24

1.7.3　数据转换　27

1.7.4　数据加载　27

1.8　数据质量　29

1.8.1　主要问题　29

1.8.2　评价标准　30

1.8.3　管理目标　31

1.8.4　管理体系　32

1.8.5　数据规划　35

目 录

1.8.6 技术方案 38
第 2 章 数据仓库设计和实现 50
2.1 数据仓库设计 50
2.1.1 设计方法 52
2.1.2 体系结构设计 53
2.1.3 数据模型设计 55
2.1.4 ETL 设计 74
2.2 数据仓库实现 80
第 3 章 数据仓库实例 84
3.1 实例一 84
3.1.1 选择主题 84
3.1.2 逻辑模型 85
3.1.3 物理模型 92
3.1.4 ETL 93
3.2 实例二 97
3.2.1 总体结构 97
3.2.2 概念模型 99
3.2.3 逻辑模型 100
3.2.4 物理模型 106
3.2.5 数据清洗 108
3.2.6 ETL 109
第 4 章 数据仓库应用——OLAP 和 OLAM 115
4.1 OLAP 115
4.2 OLAM 119
4.2.1 体系结构 120
4.2.2 特点 122
4.2.3 基于 Web 的 OLAM 123

第二篇 数据挖掘

第 5 章 数据挖掘基础 127
5.1 概述 127

目录

5.1.1 定义 127
5.1.2 功能 130
5.1.3 模型 131
5.1.4 展望 137
5.2 实现 139
5.3 工具 140
5.3.1 概述 140
5.3.2 比较 141
第6章 聚类分析 145
6.1 硬聚类 146
6.1.1 概述 146
6.1.2 相似度计算 149
6.1.3 实现方法 151
6.1.4 主要算法 152
6.2 模糊聚类 165
6.2.1 概述 165
6.2.2 主要算法 168
6.3 评价 171
第7章 分类和预测 177
7.1 神经网络 178
7.2 决策树 182
7.3 实现过程 187
第8章 关联分析 189
8.1 概述 189
8.2 Apriori 192
8.3 FP-Growth 196
第9章 Web挖掘 198
9.1 概述 199
9.1.1 定义和分类 199
9.1.2 主要技术 202
9.1.3 实现过程 213
9.2 Web资源获取 215

目 录

9.3 Web 预处理 217
9.3.1 Web 过滤 217
9.3.2 Web 去重 224
9.4 Web 抽取和表示 236
9.4.1 Web 抽取 236
9.4.2 Web 表示 236
9.5 Web 特征提取 238
9.6 Web 聚类 240
9.7 Web 分类 242
9.7.1 朴素贝叶斯 243
9.7.2 支持向量机 244
9.7.3 评价 245
第 10 章 数据挖掘实例 247
10.1 客户细分 247
10.1.1 定义 247
10.1.2 数据准备 250
10.1.3 建模过程 251
10.1.4 结果 256
10.2 重入网识别 258
10.2.1 定义 258
10.2.2 数据准备 258
10.2.3 建模过程 265
10.2.4 结果 267
10.3 虚开欺诈识别 268
10.3.1 定义 268
10.3.2 数据准备 268
10.3.3 建模过程 269
10.3.4 结果 269
10.4 数据业务收入预测 272
10.4.1 定义 272
10.4.2 数据准备 272
10.4.3 建模过程 284

目录

10.4.4 结果 286
10.5 移动客户流失预测 287
10.5.1 定义 288
10.5.2 数据准备 289
10.5.3 特征变量选取 289
10.5.4 建模过程 291
10.5.5 结果 293
10.5.6 应用 298
10.6 WAP 日志挖掘 299
10.6.1 定义 300
10.6.2 数据准备 301
10.6.3 建模过程 305
10.6.4 结果 306

第三篇 语义网和本体

第 11 章 知识基础 311
11.1 概述 311
11.2 知识分类 316
11.3 知识表示 316
11.3.1 知识表示观 317
11.3.2 知识表示方法 319
11.4 知识可视化 325
11.4.1 主要技术 326
11.4.2 工具 333
11.5 知识管理 335
11.5.1 概述 335
11.5.2 模型和技术 338
11.5.3 知识管理系统 341
11.5.4 方法和步骤 343

目录

第 12 章　语义网和本体　345
12.1　语义网　345
12.1.1　概述　345
12.1.2　层次结构　349
12.1.3　元数据　351
12.1.4　核心技术　353
12.1.5　开发工具——Jena　356
12.1.6　Web 3.0　356
12.2　本体　358
12.2.1　哲学本源　358
12.2.2　定义　359
12.2.3　建模　359
12.2.4　分类　360
12.2.5　构建方法　360
12.2.6　描述语言　363
12.2.7　实例　365
参考文献　372

第一篇

数据仓库

第1章　数据仓库基础
第2章　数据仓库设计和实现
第3章　数据仓库实例
第4章　数据仓库应用——OLAP和OLAM

第1章　数据仓库基础

1.1　概述

人类进入信息时代以来，特别是近些年，数据规模日益扩大，数据呈爆炸式增长。图灵奖获得者吉姆·格雷曾提出一个经验定律，即网络环境下每18个月产生的数据量等于有史以来的数据量之和，仅仅依靠数据库管理系统的查询检索机制和统计分析方法，已经远远不能满足实际需求，面临着“数据爆炸，知识匮乏”的严峻挑战。例如股票经纪人需要从日积月累的大量股票行情变化的历史记录（数据）中发现其规律以预测未来的趋势；天文学家需要从天文望远镜获取的观测数据（其规模可达数千GB）中发现新的遥远天体及其运动规律；医生需要从大量病人的电子病历中发现某种疾病的起因、症状等。这些数据的共同特点是：其一数据量巨大，一般都是GB乃至TB级；其二均以结构化的形式存储在数据库中，包含了大量潜在、有价值的知识，有的已被发现，有的还未被发现。如何有效地管理和利用这些海量数据？如何发现其中潜在的知识？这需要一种新的、更为有效的手段对各种数据进行整合并挖掘以发现新知识，更好地发挥这些数据的潜能。因此，数据仓库（Data Warehouse，DW）和数据挖掘（Data Mining，DM）技术应运而生。

数据仓库是一个可更好地支持企业或组织决策，面向主题的、集成的、相对稳定的、随时间不断变化的数据集合；数据挖掘则是利用计算机对海量数据进行快速、有效地分析和处理，从中获取知识，并以一种形式化的、可以理解的方式表达，以便于决策的过程。目前，数据仓库和数据挖掘技术已经成为计算机领域的研究热点之一，引起了知识发现、机器学习和统计分析等领域专家的广泛关注。

1.1.1　演变

数据仓库是建立在传统事务型数据库基础之上，为企业决策支持系统（Decision Support System，DSS）及数据挖掘系统提供数据源。到目前为止，国外数据仓库已经发展了二十几年的时间，国内虽然起步较晚，但发展较为迅速。目前已有众多的大型公司或企业正在建或已经建设不同规模的数据仓库。

传统数据库（普通数据库）和数据仓库的最根本区别在于其侧重点的不同。数据处理分为两大类：事务型处理，又称联机事务处理（Online Transaction Processing，OLTP）；分析型处理，又称联机分析处理（Online Analytical Processing，OLAP）。事务型处理以传统的数据库为中心进行企业日常的业务处理；分析型处理以数据仓库为中心分析数据背后的关联和规律，为企业决策提供可靠、有效的科学依据。事务型处理和分析型处理的分离，划清了数据处理的分析型环境与事务型环境之间的界限。从而由原来以单一数据库为中心的数据环境演变为以数据库为中心的事务处理系统和以数据仓库为基础的分析处理系统。企业

的生产环境也从以数据库为中心发展为以数据库和数据仓库为中心。因此,在事务处理环境中直接构建分析处理应用是不适合的,要提高分析和决策的效率和有效性,分析型处理及其数据必须与操作型处理及其数据相分离,必须把分析型数据从事务处理环境中提取出来,按照决策支持的需要重新组织,建立相对独立的分析处理环境,数据仓库正是为了构建这种新的分析处理环境而出现的一种数据存储和组织技术。

传统数据库的主要任务是进行事务处理,所关注的是事务处理的及时性、完整性和正确性,而在数据分析方面,则存在诸多不足,主要体现在缺乏集成性、主体不明确和分析处理效率低等多个方面。

1. 缺乏集成性

首先,企业数据库系统与部门条块分割,导致数据分布分散化与无序化。在一个企业内部,生产、销售和财务等部门往往各自使用一套满足自身工作需要的应用程序。各个部门的应用系统往往不能共享数据,缺乏数据的统一管理和维护。尽管企业内部拥有的数据量庞大,但各自封闭,构成相互独立的所谓“信息孤岛群”,无法形成统一体。其次,业务数据库缺乏统一的定义与口径,导致数据定义存在歧义。

2. 主题不明确

建立传统数据库的目的是为了满足事务处理的需要,数据库和表的定义与设计完全以此为基础。而对于数据分析而言,这些库和表无疑缺少明确的主题。

3. 分析处理效率低

设计基于传统数据库的应用系统的核心准则是保证事务处理及时而准确。显然,处理大量分析型数据的效率得不到保证。

数据仓库是因为用户需求增加而对某一类数据库应用范围的界定。仅从数据存储容器的角度而言,数据仓库与数据库并没有本质的区别。且在很多时候,数据仓库是被作为一个数据库应用系统来看待的。因此,不应该说数据库到数据仓库是技术的进步。

一般地,数据仓库是在传统数据库的基础上发展起来的,建立在异构业务数据库的基础上。尽管传统数据库对处理分析型数据存在缺陷,但数据仓库并不是对数据库的彻底抛弃。两者存在诸多差别,如表 1.1 所示。

表 1.1 数据库与数据仓库的区别

	数据库	数据仓库
内容	与业务相关的数据	与决策相关的数据
数据模型	关系、层次结构	关系、多维结构
访问	经常是随机地读、写操作	经常是只读操作
负载	事务处理量大,但每个事务涉及的记录数很少	查询量小,但每次需要查询大量的记录
事务输出	一般很少	可能非常大
停机	可能意味着灾难性错误	可能意味着决策延迟

从数据库到数据仓库的演变过程如图 1.1 所示。

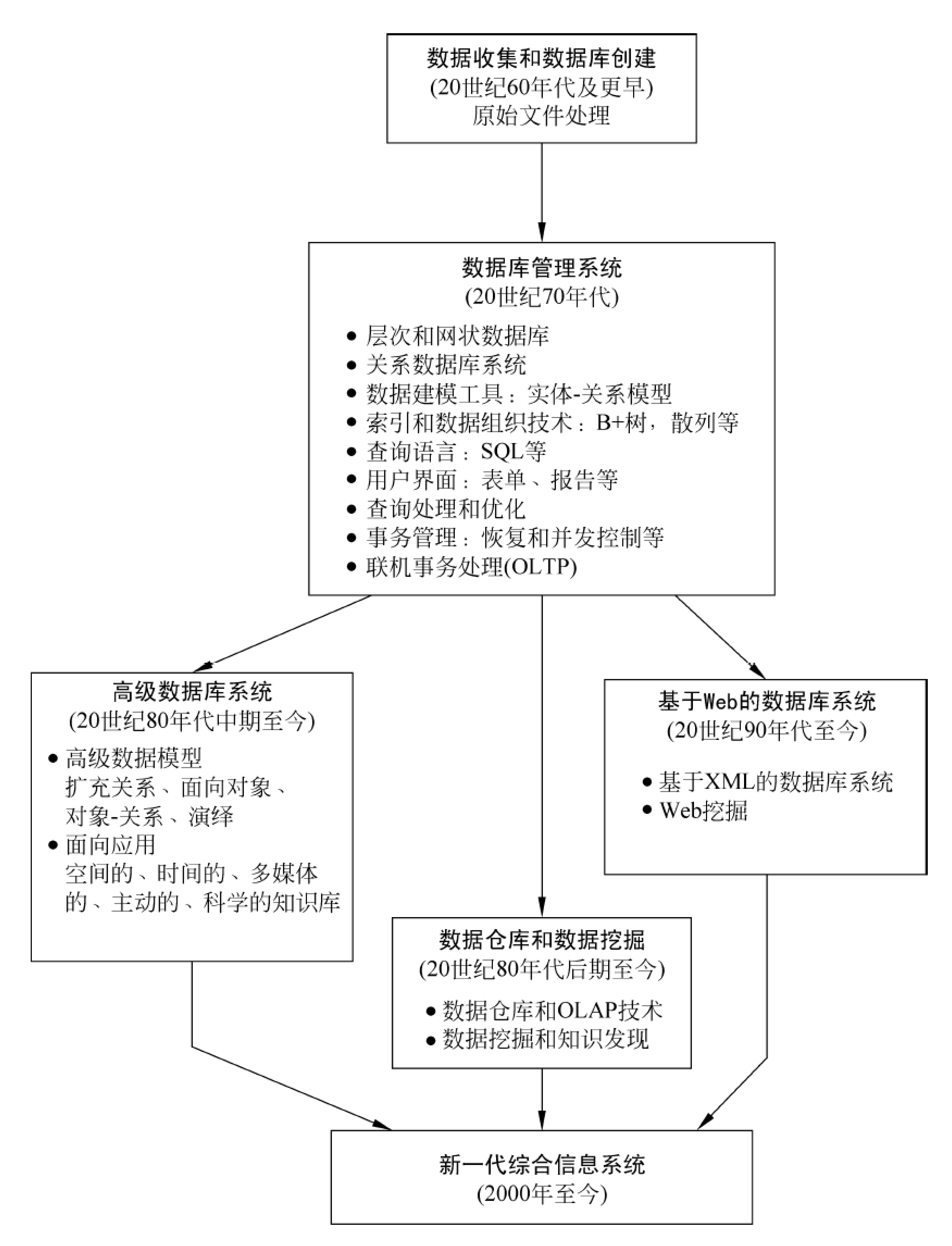


图 1.1　数据库到数据仓库的演变过程

1.1.2　定义

数据仓库的概念最早出现在 20 世纪 80 年代。1993 年，被称为“数据仓库之父”的 William H. Inmon 首次系统地阐述了数据仓库定义，即一个面向主题的、集成的、不可修改的且随时间变化的数据集合，以支持管理人员的决策。

面向主题是相对于传统数据库的面向应用而言。所谓面向应用，是指系统实现过程中主要围绕着一些应用或功能，而面向主题则是考虑一个个的问题域，对问题域涉及的数据和分析数据所采用的功能给予同样的重视。

数据仓库是面向在数据模型中已定义业务的主要主题域的，例如在电信领域中典型的主题域包括客户、产品、资源、渠道、服务和竞争等。

集成是指数据仓库中的数据来自不同的数据源。由于历史的原因,各数据源的组织结构往往不同,在这些异构的数据导入到数据仓库之前,需经历一个集成过程。在数据仓库的所有特点中,这是最重要的。应用系统的设计人员历经多年制定出来的不同的设计策略有很多种不同的表示方法,在编码、命名习惯、属性和属性度量等方面往往是不一致的。当数据导入数据仓库时,需要采用某种方法来消除应用系统中存在的不一致性。例如"客户性别"的编码,在数据仓库中是编码为"男/女"还是"m/f"并不重要,重要的是无论是什么原始应用系统,在数据仓库中应该编码一致。如果应用系统中编码为"X/Y",则在其导入数据仓库时就应进行转换。对所有的应用都要考虑一致性,如命名习惯、键码结构、属性度量以及数据特点等。

不可修改是指与面向应用的事务数据库需要对数据进行频繁地插入、更新操作不同的是,数据仓库中对数据的操作仅限于数据的初始导入和查询。操作型数据一般是一次访问和处理一条记录,也可以对操作型数据进行更新。但数据仓库中的数据则不然,通常是一起载入与访问,在数据仓库中并不进行一般意义上的数据更新。

随时间变化是指数据仓库以维的形式对数据进行组织,时间维是数据仓库中很重要的一个维度,并且数据仓库中数据的时间跨度较大,从几年甚至到几十年,称为历史数据。数据仓库中数据随时间变化的特性表现在以下几个方面:

- 数据仓库中数据的时间期限要远远长于操作型数据库中数据的时间期限。操作型数据库中数据的时间期限一般是60~90天,而数据仓库中数据的时间期限通常是5~10年。
- 操作型数据库含有"当前值"的数据,这些数据的准确性在访问时是有效的,同样当前值的数据可被更新。而数据仓库中的数据仅仅是一系列某一时刻生成的复杂快照。
- 操作型数据的键码结构可能包含也可能不包含时间元素,如年、月和日等,而数据仓库的键码结构总是包含某一时间元素。

数据仓库是DSS的基础。因为,在数据仓库中只有单一集成的数据源,并且数据是可访问的。所以与传统数据库相比,在数据仓库中DSS分析人员的工作将容易得多。

1.2 体系结构

1.2.1 两层的体系结构

由数据仓库的定义可知,它是将企业各个业务系统中与分析有关的数据集成在一起,同时数据仓库面向的应用是分析型操作,因此形成了DB-DW两层数据仓库体系结构,如图1.2所示。

其中,业务系统作为主要的分析数据来源,其数据格式主要是表的形式。实际中,由于要保证业务系统的正常运行,一般不直接在业务系统中进行数据的查询和抽取,而是采取备份库或者文件传输的方式进行数据仓库的数据抽取。外部数据源是指数据来源于企业的外部,描述企业运营的外部环境与企业经营分析有关的数据,如各个企业的市场份额等,外部数据作为经营分析的补充,对企业经营决策的正确性起着十分重要的作用,因此应保证外部数据的实时性和准确性。外部数据源具有多样性的特点,如年报等都可以作为外部数据源,同时外部数据源的格式也不统一,如文本、表格和图像等。因此对外部数据源及其数据格式

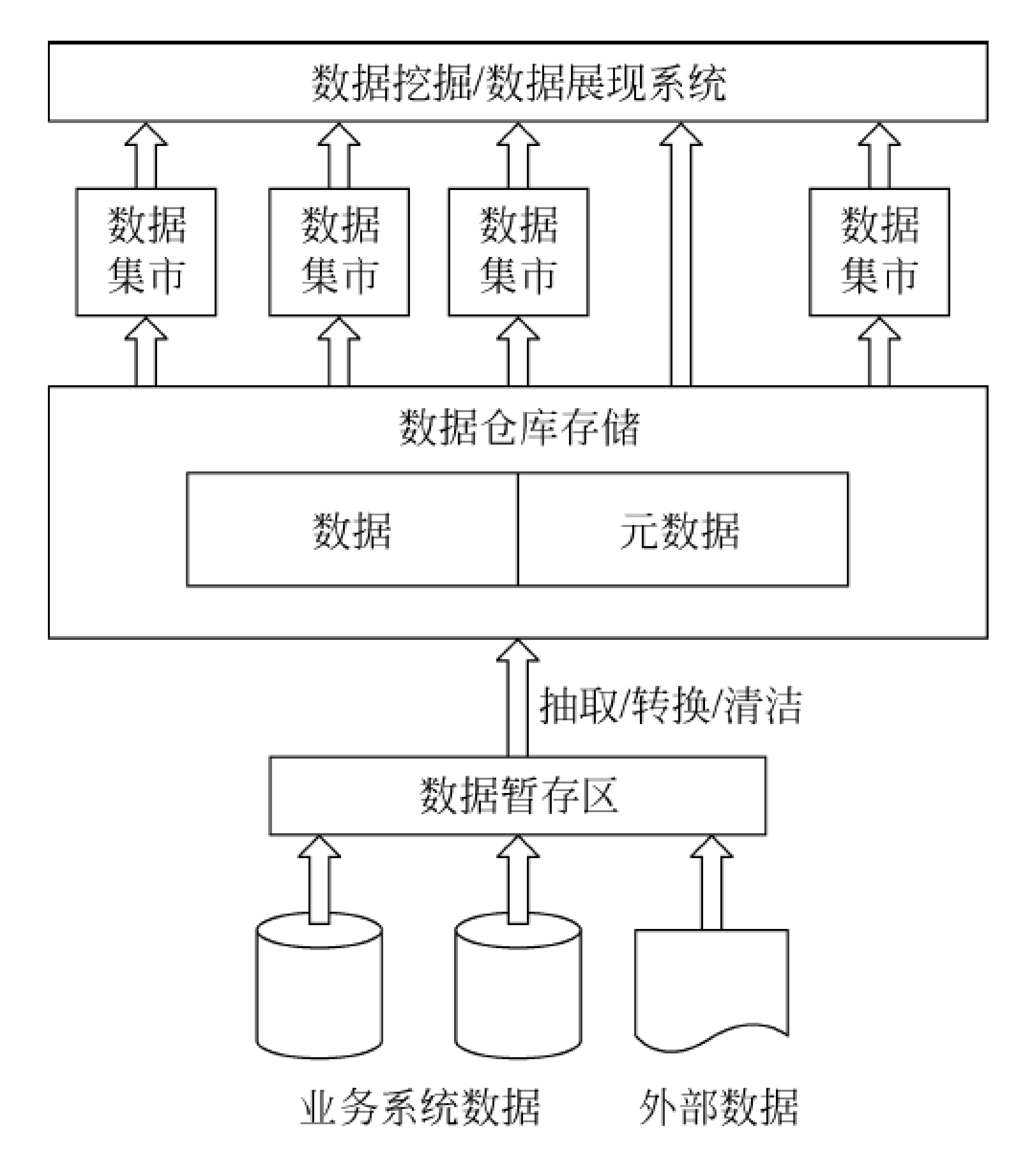


图 1.2　DB-DW 两层体系结构

等都要在数据仓库的元数据中进行记录，同时元数据中还应对外部数据的可信程度有一定评价。

由于数据仓库中的数据源不统一，同时源数据的存储形式也不相同，因此有必要在数据进入数据仓库前先将数据存放在一个统一的暂存区中，引入数据暂存区的主要作用在于：

- 统一不同数据源的数据格式，将不同数据源中不同的数据格式转换成统一的数据格式，供数据仓库统一处理。
- 进行数据的初步检查，在数据进入数据仓库之前，先对数据进行初步检查，鉴于不影响数据仓库的处理时间，这里的检查将仅涉及比较粗略的数据检查，如记录数量、关键字段是否丢失等，对于错误的数据暂不导入数据仓库，这样对进入数据仓库的数据质量有一定的保证，但是更复杂的数据清洁工作，如字段格式的统一以及数据内容的清洗这种单一记录级的处理工作则应该在数据抽取时完成。

数据暂存区可以多种存储形式实现，如文件目录或者数据库表。

数据仓库中保存了大量的历史数据，同时数据仓库面向的是整个企业的分析应用，但在实际应用中不同部门的用户可能只使用其中的一部分数据，从处理速度和效率的角度出发，可以将这部分数据在逻辑或物理上进行分离，使用户无须到数据仓库的海量数据中查询，只在与本部门有关的数据子集上操作，这样就形成了数据集市(Data Mart)的概念，它是指面向企业中的某个部门(主题)在逻辑上或物理上划分出来的数据仓库的数据子集。将数据仓库按照数据的应用划分为多个数据集市，有利于数据仓库的负载均衡，保证应用的执行效率。同时，由于数据集市具有统一的数据来源——数据仓库，遵循统一的数据模型，保证了各个不同数据集市中数据的统一。

可以看出 DB-DW 两层的数据仓库体系结构是一种管道过滤器的结构，数据从数据源进入数据仓库到展示给最终用户，都有一定的关联关系，因此要保证数据仓库中数据处理的合理调度，则需要通过数据仓库的元数据完成。

1.2.2 三层的体系结构

数据仓库的提出使得操作型处理和分析型处理得以分离,从而形成了 DB-DW 两层的体系结构,但是在企业的业务处理中存在介于操作型和分析型之间的需求,需要对短期的历史数据进行分析,同时要求较快的响应速度,这种分析无法在操作型数据库中完成,因为其保存的是数据的瞬态信息,但是如果通过数据仓库完成,由于数据仓库保存了大量的历史数据,在响应时间上无法满足要求,因此提出了操作型数据存储(Operational Data Store,ODS)的概念,ODS 数据可以概括为面向主题的、集成的、可变的和当前的或接近当前的数据。其中,面向主题和集成的特点与数据仓库的概念相似;“可变的”是指 ODS 数据可以联机改变,包括增加、删除和更新等操作;“当前的”是指数据在存取时刻是最新的;而“接近当前”是指存取的数据是最近一段时间得到的。

面向主题和集成的特点使得 ODS 数据在静态特征上很接近数据仓库的数据,但是 ODS 和数据仓库之间存在重要的差别,主要体现在:

- 数据的内容不同。数据仓库中历史数据是指长期保存并可重复查询的数据,既保存细节数据,也保存综合数据。而 ODS 一般只保存细节数据,而且 ODS 数据是可以更新的,即变化的,ODS 中保存的历史数据也是近期的。
- 数据量不同,ODS 保存的数据量要远远小于数据仓库的数据量。
- 面向的应用不同。数据仓库用于长期的趋势分析或决策支持,而 ODS 主要支持企业的全局 OLTP 和即时(up to the second)决策分析应用。

引入 ODS 后,原来的 DB-DW 两层体系结构将被扩展为 DB-ODS-DW 三层体系结构,如图 1.3 所示。

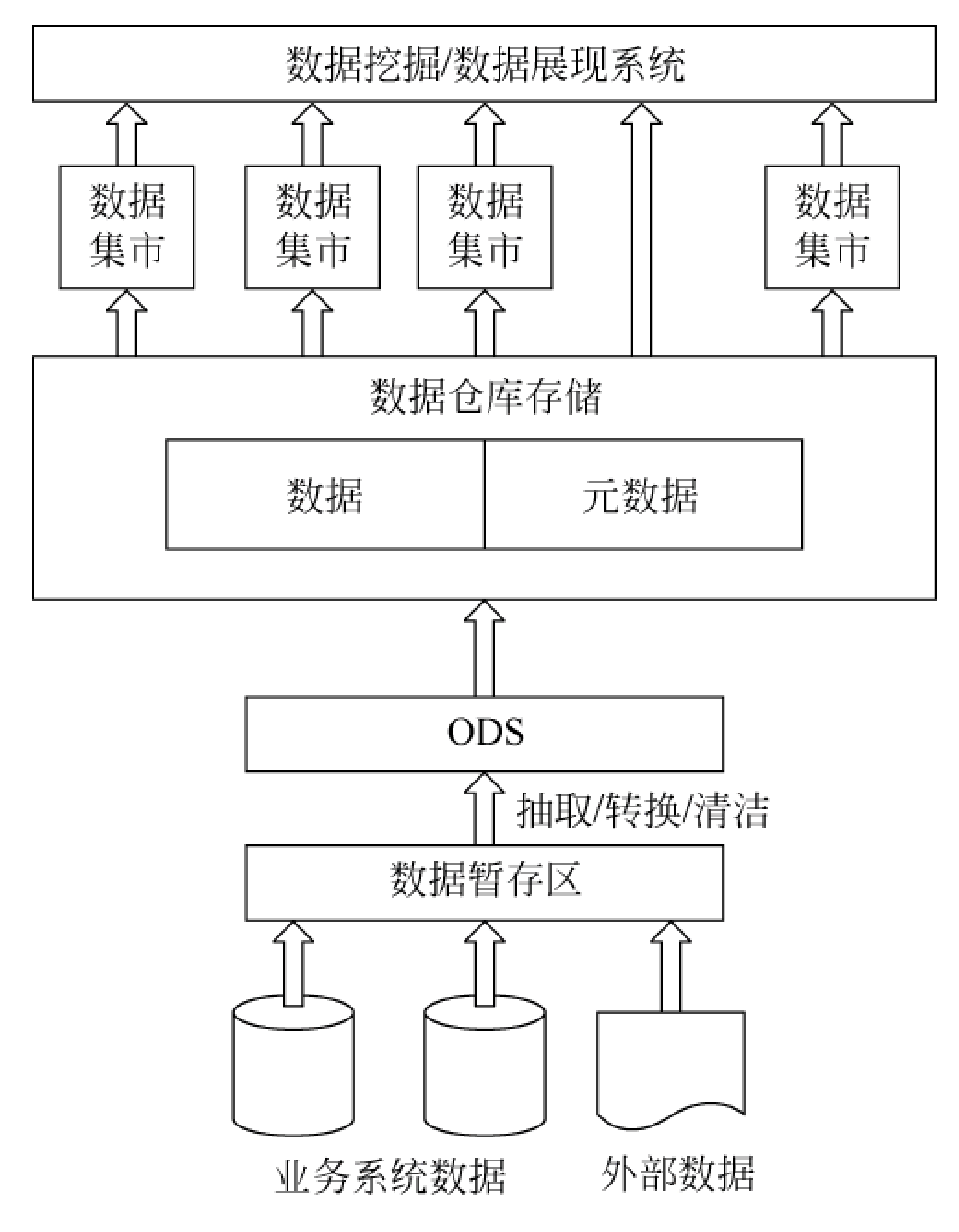


图 1.3 DB-ODS-DW 三层体系结构

在 DB-ODS-DW 三层的体系结构中，ODS 的作用可以概括为：

- 为数据仓库提供数据，减少数据仓库数据抽取的复杂性。由 ODS 的定义可知，它具有面向主题和集成两个特点，因此来自业务系统的源数据首先进入 ODS，在进入 ODS 时完成数据清洁和集成的工作，这样再向数据仓库提供的数据就是清洁的和统一的，减轻了数据仓库中数据抽取的工作。
- 即时的 OLAP 分析。由于在业务系统中需要对近期或当前的数据进行分析，如果该应用放在数据仓库中完成，由于数据仓库相应的处理环节较多，同时数据仓库保存了大量的历史数据，如果要完成这些需求势必造成留给数据仓库的数据处理时间减少，不利于数据仓库的建设，所以将这部分任务分配给 ODS，由于 ODS 保存了近期的数据，可以完成用户的即时分析需求。
- 全局的 OLTP 操作。由于 ODS 数据的集成性，整合了企业中不同业务系统的数据，同时 ODS 数据是可更新的，因此 ODS 可以提供面向企业全局的 OLTP 操作。

1.3 组成

数据仓库的大小一般都是在 100GB 以上。传统的关系型数据库是针对 OLTP 的，并不适用于数据量大且复杂度高的数据仓库。因此，数据仓库系统必须能够满足：

- 扩充数据仓库的数据。
- 每日对数据仓库系统的管理和维护。
- 允许用户增加需求。

要充分满足上述三点并不容易，尤其是数据仓库最终以自动的数据分析处理为目标。扩充数据仓库数据是非常重要的，其重点是抽取、整理并转换数据以及以适当的方式展现给用户分析使用。数据仓库并不是一个只读系统，虽然说事实数据导入数据仓库之后就不会再更新，但是如果用户需求改变，例如希望以不同的方式浏览相同的分析结果，则仍然会修订索引数据。每日对数据仓库系统的管理和维护与传统的 OLTP 系统完全不同，因为数据仓库的数据量远比 OLTP 系统大得多，所以需要更积极的管理方式。例如添加或删除数据、将数据仓库数据存入备存介质、由备存介质加载数据等。因此可以这样理解，数据仓库是一个持续更新的系统，以满足用户新的决策需求。允许用户增加需求的能力似乎是设计数据仓库系统时最困难的工作，因为每个用户都有各种不同的需求。数据仓库除了能够允许更新现有的需求之外，还应该可以增加新的分析主题。

数据仓库系统应该具有以下功能：

- 抽取数据与加载数据。
- 整理并转换数据为数据仓库适用的格式。
- 备份与备存数据。
- 管理所有的查询，将它们导向适当的数据源。

数据仓库系统的组成如图 1.4 所示，其中包括数据、信息和知识三个层次。

具体来说，数据仓库系统的管理器可细分为三种，即：

- 加载管理器，负责抽取并加载数据，在加载数据之前与过程中执行简单的转换。

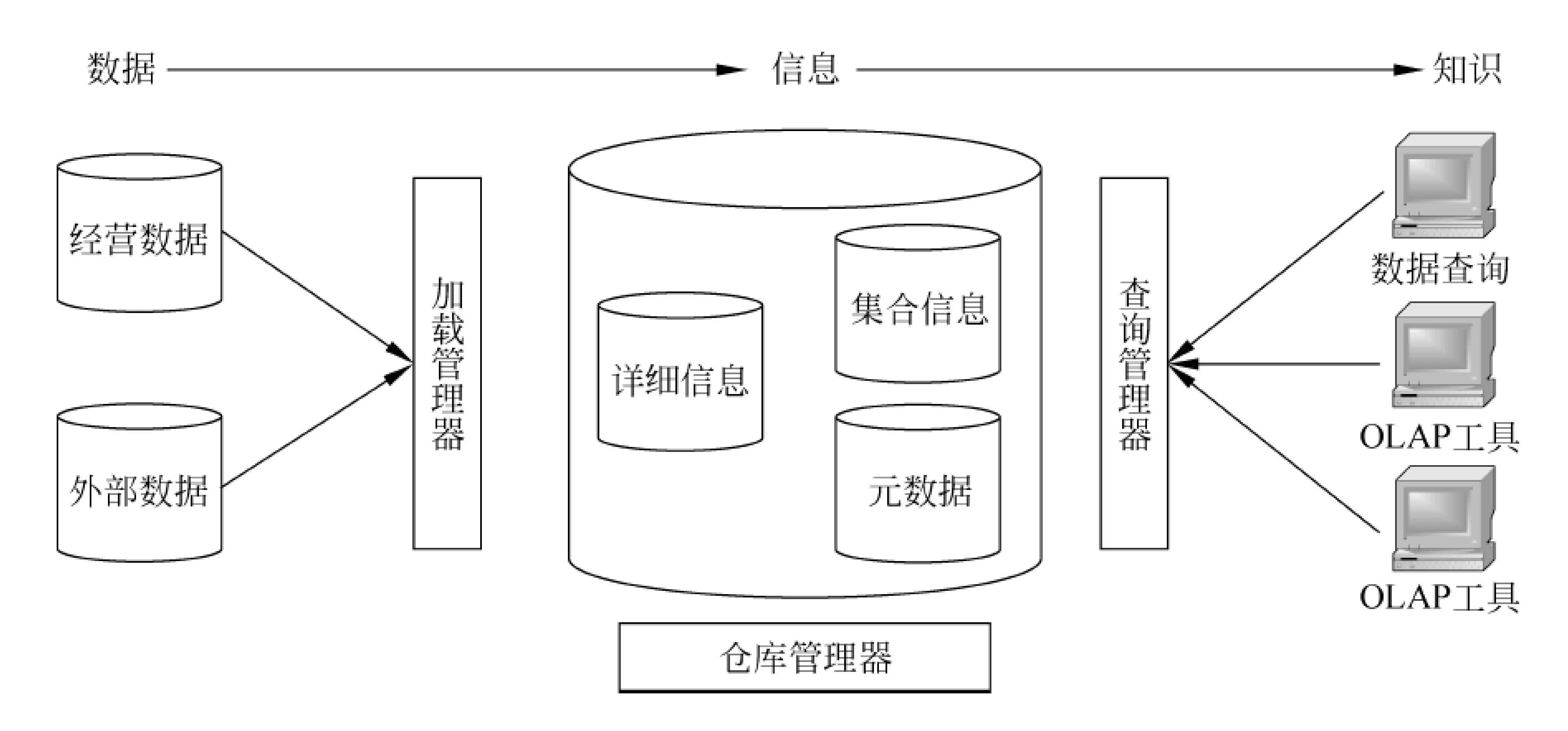


图 1.4　数据仓库系统的组成

- 仓库管理器,负责转换并管理数据仓库数据、备份与备存数据。
- 查询管理器,负责引导并管理数据仓库的查询。

1.3.1　加载管理器

加载管理器主要是支持数据抽取和加载,可以由一些外购的软件工具、针对特殊需要而编写的程序、存储过程以及脚本文件组成。用户可以尽量选用适合的软件工具来协助进行整个加载管理工作,但是因为加载管理器是每个数据仓库中功能最无法正规化的部分,会因原始数据的特性而存在很大差异,所以不可能完全使用外购的软件工具,必须自行编写针对特殊需要的程序、存储过程或脚本文件。

加载管理器应该具有以下功能:

- 自源系统抽取数据。
- 将抽取的数据快速加载到临时存储介质。
- 执行简单的数据转换。
- 将转换后的数据加载至与数据仓库类似的数据结构之中。

加载管理器的构成如图 1.5 所示。

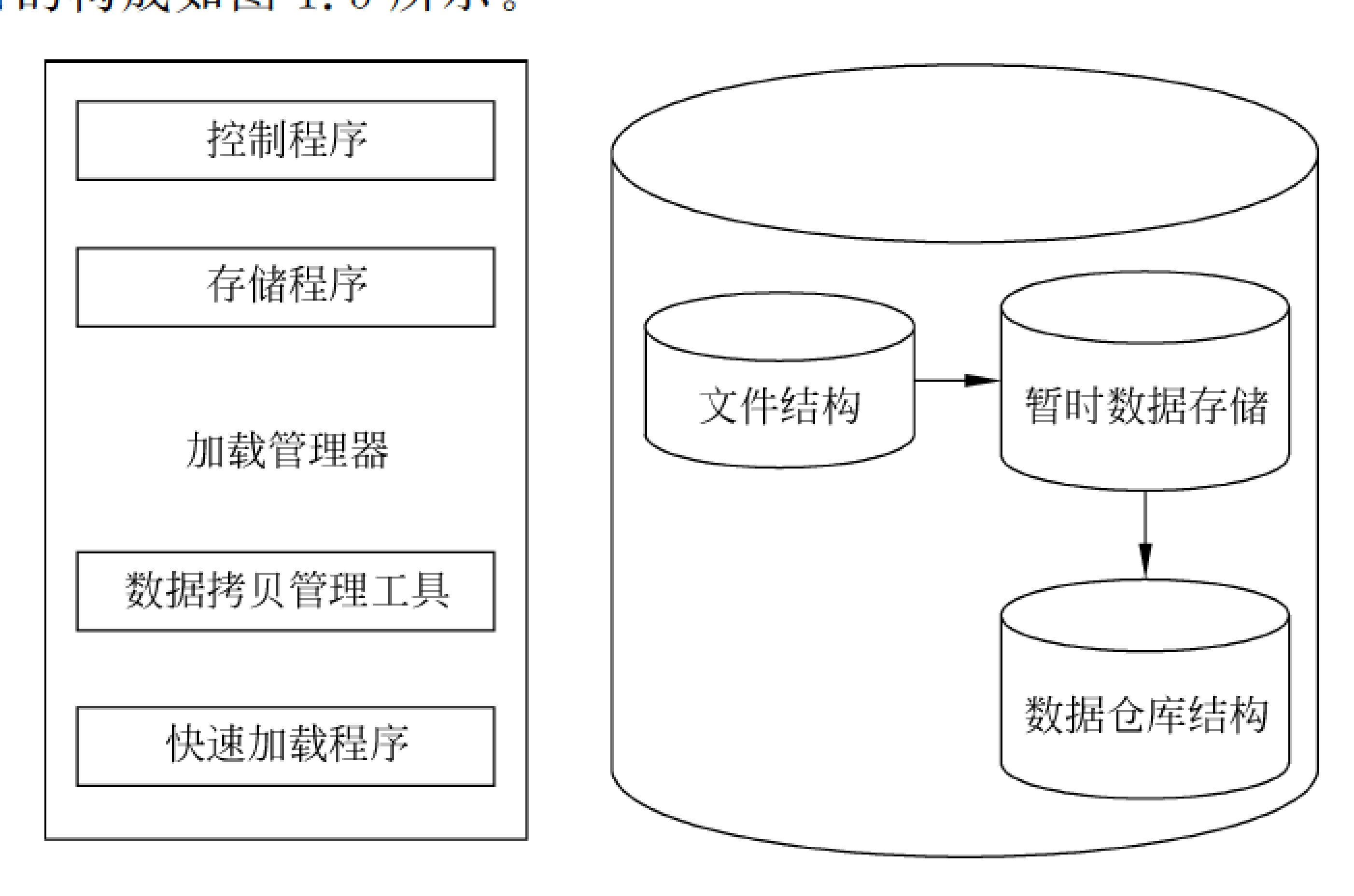


图 1.5　加载管理器的构成

上述所有功能都应该自动执行，至于数据错误的修正，也应该尽量避免人工介入。

每一种关系数据库产品都会提供数据快速加载工具，至于数据复制管理工具则会提供某种程度的转换功能。如果数据源需要比较复杂的转换，可以使用 C/C++或存储过程编写转换程序，至于工作控制流程则可以使用操作系统提供的功能或编写控制用的脚本文件。

因为数据仓库拥有的数据量非常庞大，所以需要快速的加载工具。一般而言，可以将原始数据先加载到关系数据库，然后再进行数据校验。

当原始数据已经导入临时存储介质后，可以利用数据库本身的功能与外购的软件工具进行简单的转换，这些转换不包括复杂的逻辑运算也不会使用到关联操作。当简单的数据转换执行完毕，可以使用自行设计的软件工具进行与数据仓库相关以及较复杂的转换。下面列出了一些可以在本阶段执行的数据转换与校验功能：

- 删除在数据仓库中不必要的字段。
- 将所有的数值转换为所需的数据类型。
- 将每个字段转换为正确的格式（例如大写字母、删除前置空格符等）。
- 根据企业需求校验字段值是否有效。
- 检验数据仓库所需字段是否有数据。

1.3.2 仓库管理器

仓库管理器执行管理一个数据仓库所有的必要程序，可以由一些外购的系统管理工具、针对特殊需要而编写的程序以及脚本文件组成，仓库管理器的复杂度会因自动化的程度而异。仓库管理器的构成如图 1.6 所示。

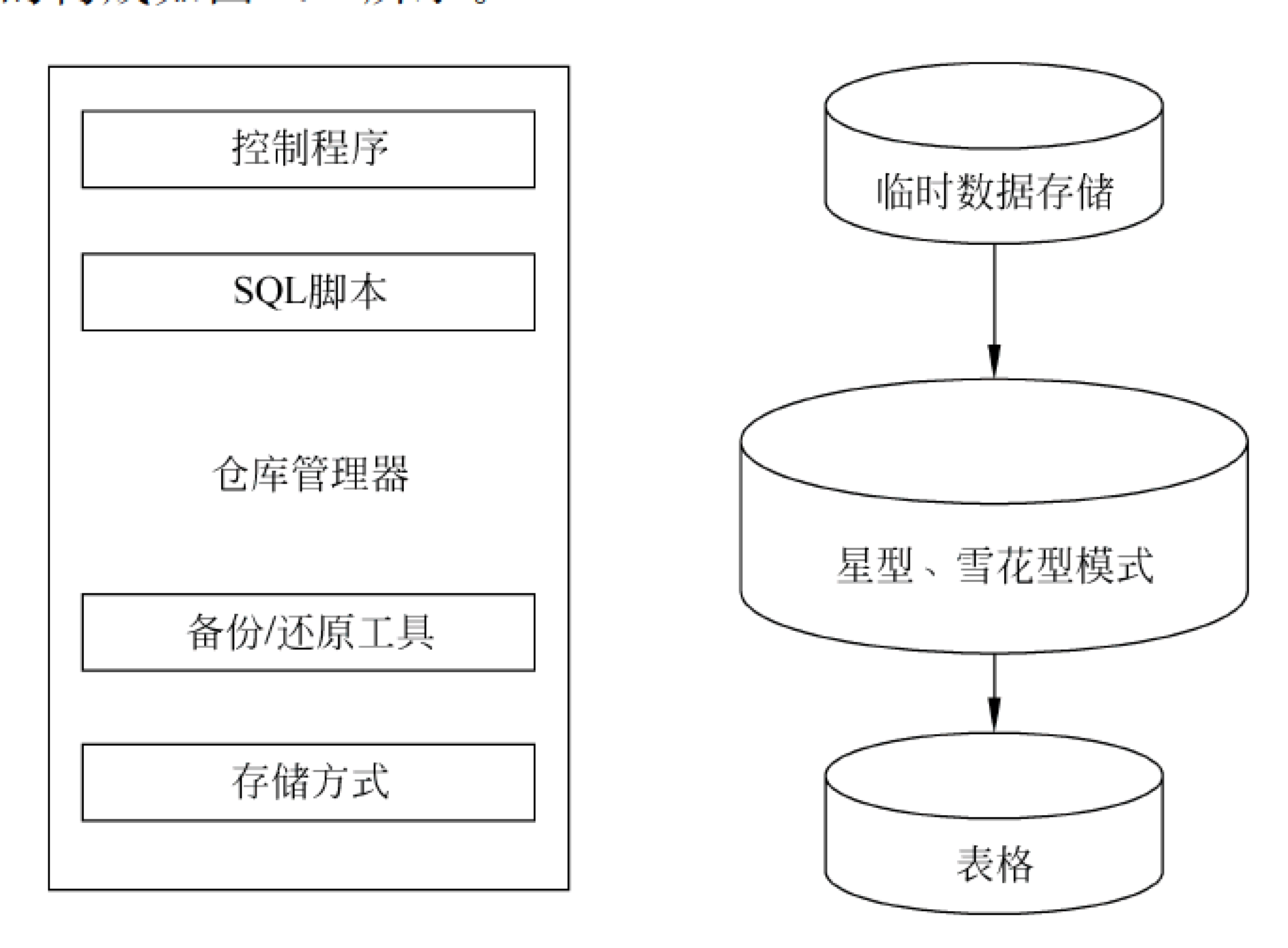


图 1.6 仓库管理器的构成

仓库管理器将执行以下功能：

- 检验各字段相互之间的关系与一致性。
- 将临时存储介质中的数据进行转换与合并，然后加载到数据仓库。
- 对数据仓库数据添加索引、视图和分区。

- 根据需要将数据进行正规化。
- 根据需要生成新的集合信息。
- 更新已有的集合信息。
- 备份数据仓库(完整或递增式)。
- 备存数据仓库中过时的数据(移存至其他存储介质)。

用户可以使用 C/C++或存储过程将数据导入到数据仓库,添加索引、视图和数据分区,生成星型、雪花型数据模式,生成、更新集合信息;可以使用操作系统与数据库管理系统提供的工具备份数据仓库、备存过时数据;也可以使用数据仓库系统提供的分析工具分析查询概述文件。

在数据仓库中可以将数据分为若干个数据分区,以便于管理。用户可以根据数据仓库中数据的更新频率创建数据分区,如果数据仓库周期以季度为单位,则可以在每一季度结束后将加入的历史数据创建成以季度为单位的数据分区,而且可以定期将小的数据分区并入大的数据分区中。

将原始数据整理、检验完毕后,仓库管理器会把数据转换为一种适用于查询管理决策信息的数据结构。

数据仓库的数据是由事实数据和维度数据组成的,事实数据是能够反映过去事实的数据,而维度数据则是为了使查询更加快速而创建的索引数据。就数据仓库的数据结构而言,是以事实表为中心,各个维表位于四周而形成的一个星型模式。为了便于快速查询,仓库管理器会为事实表和维表创建索引。事实表拥有非常多的记录,当索引文件大到一定程度时,将索引以一笔一笔的方式添加至已有的索引中,不见得是一种好方法。如果事实表包含了很多记录,建议先删除现有的索引,然后再重新创建索引,这样运行效率会较高。维表的数据量虽然也不小,但是比起事实表还是小得多。一般而言,维表不会做太多的更新,除非是维表整个更新,一般不需要删除维度数据再重新创建索引。出于管理方面的考虑,可以将事实数据分为多个数据分区,但是对用户而言,数据分区则是完全透明的。为了使用户看到的是一个独立完整的事实表,仓库管理器创建一些视图将整个数据分区合并为一个单表。

如果数据仓库周期以季度为单位,可以考虑创建以下视图:

- 为上半年创建一个视图。
- 为当年创建一个视图。
- 为上一年创建一个视图。

创建视图将降低查询的运行效率,所以建议在一个视图中不要包含过多的数据分区,而且不要创建多层视图。只要不创建包含过多数据分区的视图,系统就可以自行消化掉这些工作的开销。当数据已经完全加载了数据仓库的星型、雪花型模式之后,仓库管理器将创建一系列的集合,以加快常用的一般性查询的运行速度。因为每个查询都是在抽取某一维度的一个部分数据集合,所以仓库管理器会以此为基础决定要计算哪些集合信息。例如上一季度的销售量、某一部门的销售量、上半年整个公司的销售量。

仓库管理器的重要任务之一是管理查询概述文件,为了统计数据仓库系统内常用的查询,查询管理器搜集所有的查询并加以统计分析,归纳出为哪些常用的查询创建集合信息。

查询概述文件在数据仓库中是一种元数据,主要描述一个数据仓库中所有查询的特性。当用户更新了其需求之后,仓库管理器将更新对应的查询概述文件,然后重新生成新的集合

信息。可以使用存储过程或嵌入式 C 编写计算集合信息的应用程序。

一般的数据库管理系统都会提供搜集查询的功能，当仓库管理器由查询管理器搜集到足够多的统计信息时，可以下面的方式执行集合信息的计算工作：

- 将 SQL 查询转换为星型的查询。
- 分析星型查询，检查使用到哪些事实数据、维度数据和集合信息。
- 决定集合信息的使用频率(有多少个查询使用一项集合信息)。
- 检查目前的查询概述文件中是否已包含该项集合信息。
- 如果查询概述文件尚未包含该项集合信息，则将其定义加入。
- 定期检验不再适用的集合信息，并将其定义从查询概述文件中删除。
- 定期根据修正过的查询概述文件重新创建新的集合信息。

当然，仓库管理器不一定需要为查询概述文件中所有的项目创建集合信息，因为创建集合信息需要占用系统资源。如果数据仓库系统的资源有限，可以根据每项的使用频率设置优先级，从而仅为某些优先级足够高的项目计算集合信息。

1.3.3 查询管理器

查询管理器执行管理数据仓库系统中所有查询的相关处理程序，可以由一些外购的存取工具、数据仓库系统所提供的系统监控工具、数据库管理系统所提供的管理工具、针对特殊需要编写的程序以及脚本文件组成。同样地，查询管理器的复杂度视数据仓库系统而定。

查询管理器的构成如图 1.7 所示。

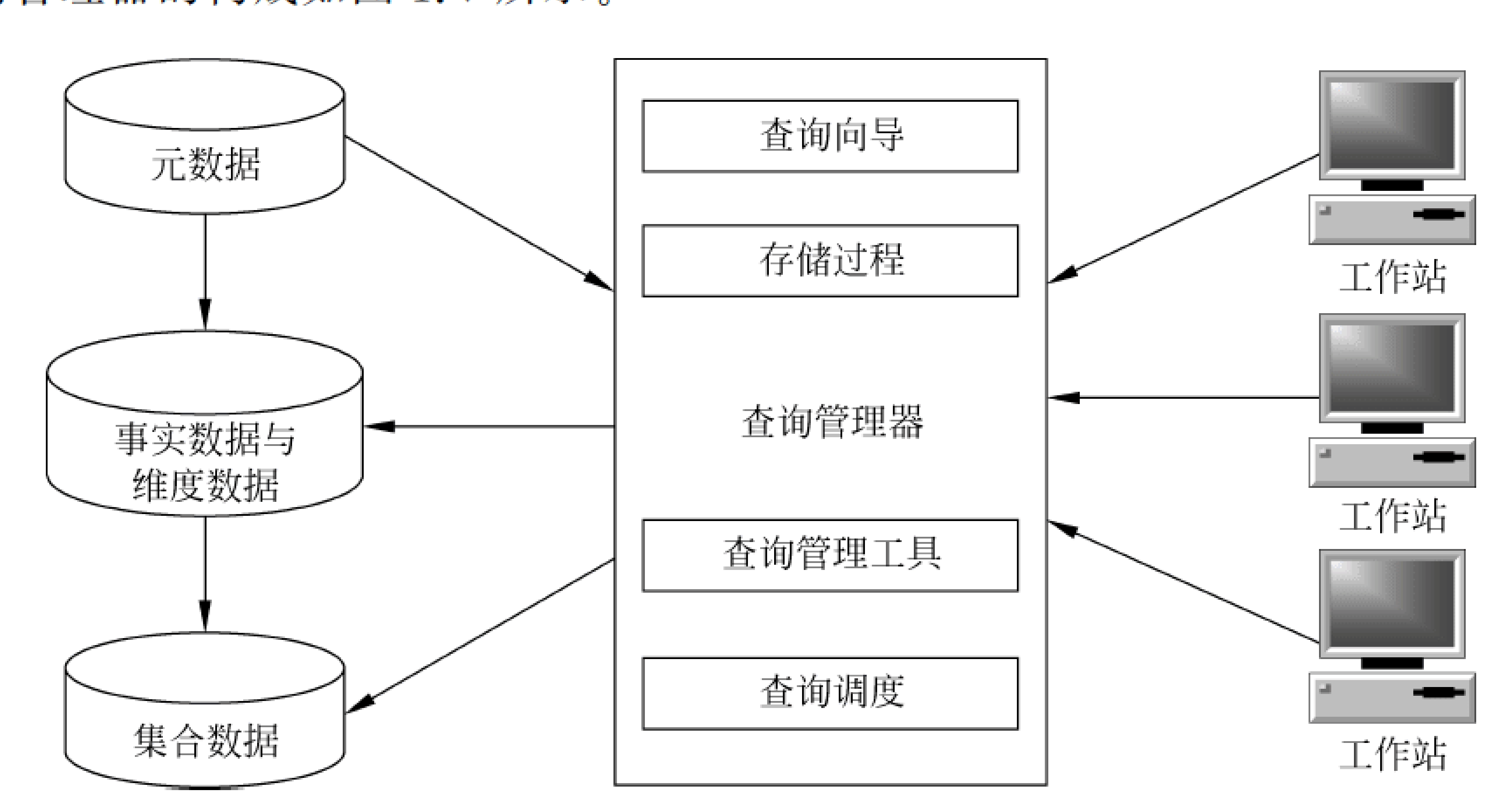


图 1.7　查询管理器的构成

查询管理器主要执行以下功能：

- 将查询引导至正确的表。
- 为所有的用户查询调度。

可以使用用户存取工具、存储过程、C/C++程序将查询引导至正确的表；也可以使用用户存取工具、存储过程、C/C++程序、数据库管理系统提供的管理工具、外购的调度软件为所

有用户的查询进行调度。

查询管理器将查询统计分析的结果存入查询概述文件,供仓库管理器使用,以决定为哪些项目执行数据集合的任务。

1.4 元数据

1.4.1 定义和分类

元数据(Metadata)是关于数据的数据。在数据仓库系统中,元数据可以帮助数据仓库管理员和数据仓库开发人员非常方便地找到他们所需的数据;元数据是描述数据仓库中数据结构和构建方法的数据。

随着计算机技术的应用日益广泛,元数据得到人们越来越多的关注,这是由多方面的需求决定的。

其一是管理数据的需求。当系统数据量越来越大时,检索和使用这些数据的效率就会降低,通过存储关于系统和数据的内容、组织、特性等细节可以帮助有效地管理,从而提高效率。

其二是系统分布、互通和重用的要求。目前信息系统一个共同的趋势是信息共享,要实现异构系统的信息共享,则需要描述数据语义以及软件开发过程的元数据,而且这些元数据必须标准化,以充分实现分布、互通和重用。

其三是元数据重用、综合的需求。目前,很少有单一工具能满足大型商业应用的需求,用户常常需要使用多种工具的组合,不同工具之间数据交换的途径之一就是通过标准的元数据。

在过去的几年里,元数据的概念在现实中大量使用,有时为了支持信息检索,有时为了软件配置,有时为了不同系统之间的数据交互。对于不同的领域专家,元数据有着不同的应用,但至少有两点是共同的,即元数据是对数据的描述;元数据的存在是为了更有效地使用数据。

对于元数据,根据观察角度的不同,可以划分为不同的类别。

按照与特定领域是否相关可划分为:

- 与特定领域相关的元数据——描述数据在此特定领域内的公共属性。
- 与特定领域无关的元数据——描述所有数据的公共属性。
- 与模型相关的元数据——描述信息和元信息建模过程的数据。此类元数据又可分为两类:

(1) 横向模型关联元数据——综合现有的两个或多个信息模型,例如两个不同数据库之间的交互、从多个数据源中提取数据时,就需要这种横向模型关联元数据。当不同的信息模型之间进行互通时,需要模型中各个层的关联描述,即横向模型关联元数据。

(2) 纵向模型关联元数据——模型信息层与元信息层之间的关联元数据。不同的层可以采用不同的模型,上层是下层的结构描述,上下层之间的对应关联,即纵向模型关联元数据。

- 其他元数据,例如系统硬件、软件描述和系统配置描述等。

按照元数据的应用场合可划分为：

- 数据元数据，又称为信息系统元数据。信息系统使用元数据描述信息源，以按照用户需求检索、存取和理解源信息。因此，数据元数据保证了在新的应用环境中使用信息，支持了整个信息结构的演进。
- 过程元数据，又称为软件结构元数据。过程元数据是关于应用系统的信息，它帮助用户查找、评估、存取和管理数据。大型软件结构中包括描述各个组件接口、功能和依赖关系的元数据，这些元数据保证了软件组件的灵活、动态配置。

按照元数据的具体内容可划分为：

- 内容（Content）——识别、定义、描述基本数据元素，包括数据单元、合法值域等。
- 结构（Structure）——在相关范围内定义数据元素的逻辑概念集合。
- 表示（Representation）——描述每个值域（多为技术相关）的物理表示，以及数据元素集合的物理存储结构。
- 文法（Context）——提供基础数据的族系和属性评估，包括所有与基础数据的收集、处理和使用相关的信息。

此外，按照用途的不同可划分为技术元数据（Technical Metadata）和业务元数据（Business Metadata）两大类。技术元数据存储关于数据仓库系统技术细节的数据，是用于开发和管理数据仓库的数据，它保证了数据仓库系统的正常运行；业务元数据是从业务角度描述数据仓库中的数据，它提供介于使用者和实际系统之间的语义层，使得数据仓库使用人员能够"读懂"数据仓库中的数据。

1.4.2 标准化

关于元数据的标准化，就内容而言大致可分为两类：一是元数据建模，即对将来元数据的组织进行规范定义，使得在元数据建模的标准制定之后产生的元数据均以一致的方式组织，从而保证元数据管理的一致性和简单性；二是元数据交互，即对已有的元数据组织方式以及相互间交互格式进行规范定义，实现不同系统元数据的交互。目前，定义元数据相关规范的主要组织机构包括：

1. 对象管理组织（Object Management Group，OMG）

1995 年 OMG 采用 MOF（Meta Object Facility）并不断完善；1997 年采用 UML；2000 年又采用通用仓库元模型（Common Warehouse Metamodel，CWM）。UML、MOF 和 CWM 这三个标准形成了 OMG 建模和元数据管理、交换结构的基础，推动元数据标准化的快速发展。

2. 元数据联合会（Meta Data Coalition，MDC）

MDC 成立于 1995 年，旨在提供标准化的元数据交互。MDC 于 1996 年开发了 MDIS（Meta Data Interchange Specification）并完成 MDC-OIM 的技术评审，MDC-OIM 基于 Microsoft 的开放信息模型 OIM（Open Information Model），是一个独立于技术、以厂商为核心的信息模型。OIM 是微软公司的元数据管理产品 Microsoft Repository 的一部分，由微软和其他二十多家公司共同开发，作为微软开放过程的一部分，经过了三百多家公司的评审。

为了推动元数据的标准化，MDC 和 OMG 在元数据标准的制定上协同工作。1999 年

4 月,MDC 成为 OMG 的成员,而 OMG 也同时成为 MDC 的成员。MDC 使用了 OMG 的 UML,而 MDC-OIM 中的数据仓库部分被用来作为 OMG 的公共仓库元数据交互(Common Warehouse Metadata Interchange,CWMI)的设计参考。在两个组织技术力量的合作努力下,元数据标准逐步实现标准化。

下面将重点介绍通用仓库元模型。

1.4.3 CWM

目前,数据仓库产品很多,它们对元数据都有自己的定义和格式,因此创建、管理和共享元数据很耗时而且容易出错。为了解决上述问题,必须采用标准的语言描述数据仓库元数据的结构和语义,并提供标准的元数据交换机制。

但是,元数据的交换涉及很多问题,例如元数据的表示形式以及交换机制等。为了解决上述问题,OMG 在 2000 年提出了一套关于数据仓库元数据的 CWM 规范,其主要目的是方便异质、分布式系统中的数据仓库工具、数据仓库平台以及元数据库之间的元数据交换,旨在推动数据仓库、商业智能和知识管理方面元数据的共享和交换。与 OMG 合作提出 CWM 规范的公司包括 IBM、Unisys、NCR、Hyperion Solutions、Oracle、UBS AG、Genesis Development、Dimension EDI,还有一些公司明确表示支持 CWM,包括 SUN、HP、Data Access Technologies、InLine Software、Aonix、Hitachi Ltd.。

提出 CWM 的主要原因在于:

- 从数据仓库开发者的角度而言,单一工具很少能完全满足用户不断变化的需求,但同时又很难对各种产品进行集成;
- 从数据仓库用户的角度而言,面对的信息量太大,无法轻易找到自己真正需要的,而且把这些信息完整正确地表示出来也是一个挑战;
- 从数据仓库供应商的角度而言,目前信息的共享还没有标准格式,元数据集成的代价太大。

CWM 主要基于以下三个工业标准:

- UML(Unified Modeling Language,统一建模语言)是 OMG 的一个建模标准。
- MOF(Meta Object Facility,元对象工具)是 OMG 关于元模型和元数据库的一个标准。
- XMI(XML Metadata Interchange,XML 元数据交换)是 OMG 关于元数据交换的标准。

这三个标准是 OMG 元数据库体系结构的核心,UML 定义了表示模型和元模型的语法和语义;MOF 为构建模型和元模型提供了可扩展的框架,并提供存取元数据的程序接口;而利用 XMI 则可以将元数据转换为标准的 XML 数据流或文件格式,便于交换,这大大增强了 CWM 的通用性。

为了说明 CWM 和这三个标准之间的关系,首先介绍一下元数据的层次。传统的元数据包括四个层次,除了最底层外,每一层都对其下一层进行描述。最底层是用户对象层(M0),包括用户描述的信息,这些信息统称为数据;上一层是模型层(M1),由描述信息的元数据组成,在这一层元数据一般都组合成模型的形式;再上一层是元模型层(M2),由定

义 M1 层元数据格式和语义的描述信息组成，即元元数据，一般组合成元模型的形式；最高层是元元模型层（M3），它定义元模型的结构和语义。四层结构的示例如图 1.8 所示。

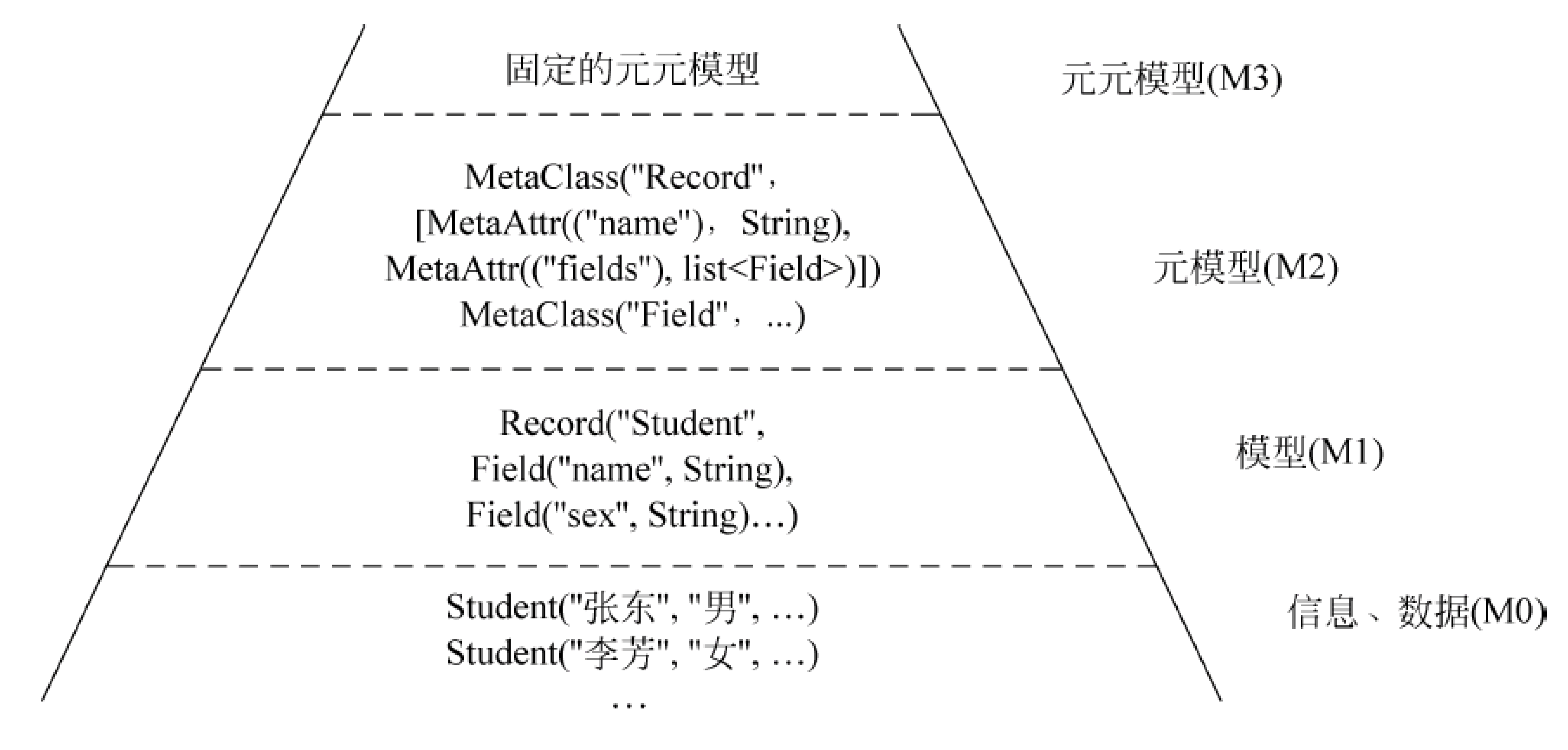


图 1.8　元数据层次结构示例

其中，四层分别是：

- 数据层是学生记录（Record）的实例，即具体的某个学生；
- 模型层描述学生这个记录类型的内容，它有一个名字（Student）和两个字段（Field），每个字段都有一个名称和类型，例如第一个字段的名称是 name，字段类型是 String；
- 元模型层对 Record 这一类型进行定义，在这一层 Record 是元类 MetaClass 的一个实例，一个 Record 拥有两个元属性 MetaAttribute：第一个 name 定义其名字，是 String 类型；第二个 fields 定义它包含的字段集，字段集的成员是 Field 类型。类似地，元类 Field 应该也包含两个元属性，即名字（Name）和类型（Type）；
- 元元模型层的结构基本固定，它将所有概念抽象为以下组件：元类 MetaClass、元属性 MetaAttribute 和元关联 MetaAssociation，并定义了元类之间的关系，主要包括包含（Contains）、继承（Generalizes）、类型引用（IsOfType）和依赖（DependsOn）。

理论上还可以再向上抽象，但由于元元模型层是自描述的，所以四层足够了。至于为什么要抽象出上面两层，则是为了支持各种不同的模型和元模型。

综上所述，OMG 规范和元数据四层结构的对应关系如表 1.2 所示。

表 1.2　OMG 规范和元数据四层结构的对应关系

元数据层次	MOF 术语	示　　例
M3	元元模型	MOF 模型
M2	元模型，元元数据	UML 元模型，CWM 元模型
M1	模型，元数据	UML 模型，CWM 模型
M0	对象，数据	数据仓库数据

1. 组成

CWM 完整地描述了数据仓库元数据交换的语法、语义以及用于异质平台之间的元数据交换机制，由三部分组成。

1）CWM 元模型

CWM 元模型描述数据仓库的组成元素，用户可以按照这些元模型开发相应的组件，例如 ETL、OLAP 和数据挖掘等。为了降低复杂度并重用，CWM 元模型采用分层的方式组织所包含的包，如图 1.9 所示。

管理包	数据仓库处理			数据仓库运行		
分析包	转换		OLAP	数据挖掘	信息可视化	业务命名
资源包	面向对象	关系	记录	多维		XML
基础包	业务信息	数据类型	表达式	关键字和索引	类型映射	软件部署
	UML1.3					

图 1.9　CWM 元模型的包结构

图 1.9 中 CWM 元模型主要包括基础(Foundation)包、资源(Resource)包、分析(Analysis)包和管理(Management)包，共四层。

基础包主要定义为 CWM 其他包所共享的一些基本概念和结构，包含的子包为：

- Business Information——定义面向业务的通用信息，例如负责人信息等。
- Data Types——定义其他包用以创建所需的数据类型的元模型组件。
- Expressions——定义 CWM 其他包定义表达式树所需的元模型组件。
- Keys and Indexes——定义描述关键字和索引的共享元模型。
- Software Deployment——描述一个软件在数据仓库中如何被使用的元模型。
- Type Mapping——支持不同系统之间数据类型映射的元模型。

资源包主要定义一些描述常用的源/目标数据的元模型，包含的子包为：

- Relational——描述通过关系型接口访问的数据库的数据模型和元模型，例如 RDBMS、ODBC 和 JDBC 等。
- Record——描述记录的基本概念和结构的元模型，这里记录的概念很广泛，它可以描述任何结构化的信息，例如数据库的一条记录、文档等。
- Multidimensional——描述多维数据库的元模型。
- XML——描述用 XML 表示的源/目标数据。

分析包主要定义一些描述数据仓库工具的元模型，包含的子包为：

- Transformation——定义数据仓库中抽取转换规则的元模型，包含对各种类型数据源之间的转换规则的描述。
- OLAP——对 OLAP 工具和应用进行描述，并定义其到实际系统的映射。
- Data Mining——对数据挖掘工具和应用进行描述。
- Information Visualization——定义问题域中有关信息发布或信息可视化的元模型。
- Business Nomenclature——对业务数据进行描述，例如业务术语及其适用范围等。

管理包主要定义一些描述数据仓库运行和调度信息的元模型，包含的子包为：

- Warehouse Process——描述数据仓库中抽取转换规则的执行过程，即各个转换规则的触发条件。
- Warehouse Operation——描述数据仓库日常运行情况的元模型。

2）CWM DTD 和 CWM XML

CWM DTD 和 XML 对应于 CWM 中所有包的 DTD 和 XML，它们都遵循 XMI 规范。定义 CWM DTD 和 XML 的主要目的是为了基于 XML 进行元数据交换。因为 XML 在各个领域的应用越来越广泛，CWM 提供元模型到 XML 的转换，无疑增加了其自身的通用性，各种分析工具和元数据库可以利用这些模板为其元模型生成 DTD 和 XML 文档，这样就可以和其他的工具之间进行元数据交换。

3）CWM IDL

CWM IDL 为上述所有的包定义了符合 MOF 1.3 的 IDL 接口，这样就可以利用 CORBA 进行元数据交换。用户可以创建一些具有分析功能的软件包，例如数据挖掘组件等，提供 CWM 中规定的 IDL 接口，就可以被其他支持 CWM 的工具和数据仓库调用，这大大增强了 CWM 的灵活性和适用性。

2. 特点

通过对 CWM 组成的介绍，可以看出 CWM 具有以下特点：

- 对所有的数据仓库功能元数据定义了详细的元模型和交换方式，包括技术元数据（例如 Software Deployment、Transformation、Warehouse Process 等）和业务元数据（例如 OLAP、Business Information 等）。
- 定义了一个通用且强大的 Transformation 包，可以表示任何源数据和目标数据之间的转换规则。此外，还为多种常用的源/目标数据（例如 Relational、Record、Multidimensional、XML 等）和工具相关的数据源（例如 IMS、DMSII、COBOL Data、Essbase 和 Express 等）定义了元模型和交换方式。
- 对所有的数据仓库运行元素定义了元模型和交换方式，包括调度、状态报告和历史记录等。
- 对所有的分析型数据以及主要的分析型数据模型定义了元模型和交换方式，例如多维模型。
- 对操作型数据以及主要的操作型数据模型定义了元模型，例如关系型和面向对象型。

3. 目标和原则

CWM 的主要设计目标和原则包括：

- 对 UML 中概念的重用——UML 1.3 是整个 CWM 的设计基础，CWM 在任何可能的地方对 UML 中的概念进行重用，所有的 CWM 对象类型都直接或间接地继承于 UML，因此也继承了它们的属性和方法。这样可以节省很多重复工作，并且使 CWM 更容易理解，所有熟悉 UML 的用户都可以有一个比较高的起点。
- 模块化——CWM 元模型被分成许多包，以便它们分别实现并减少复杂度。
- 通用化——CWM 元模型独立于任何具体的数据仓库工具。同时，它尽量多地包含了基于特定工具实现的有代表性且通用的数据仓库特点。换言之，只有那些多种工具共享的信息才会被 CWM 元模型包含进来。

4. 应用

CWM 主要面向以下几类用户：

- 数据仓库平台和工具提供商——CWM 提供了一个组件可插卸的通用系统框架，因

为这是一种全球通用的元数据交换协议,所以可以很方便地在各种异构平台上发布自己的产品。

- 数据仓库服务提供者——可重用、可编辑、可扩展的 CWM 元数据大大提高了工作效率。因为 CWM 与产品无关,所以可以避免大量的重复工作。
- 数据仓库管理员——数据仓库管理员有时需要对现有工具进行整合,而 CWM XML 无疑提供了一种最方便的整合方式。另外,管理员经常需要对资源进行增减、分区或者重新分配,CWM 提供了这方面的元数据以帮助他们完成这些工作,并对改变造成的影响做出评估。
- 终端用户——CWM 为查询和展示工具定义了元模型,更方便、快捷地为其展示所需的信息。
- 信息技术管理者——CWM 为系统管理和报表工具定义了元模型,使得用户能够更轻松地对系统和信息进行管理。

1.4.4 UML、MOF 和 XMI 与 CWM 的关系

1. UML 与 CWM

UML 是一种面向对象的建模语言,由曾经是面向对象软件建模三种主流语言的 Booch、OMT 和 OOSE 综合而得,后来被 OMG 定义为面向对象建模的标准语言。目前有很多图形工具支持它,并已得到广泛应用。

UML 定义了多种模型元素,支持面向对象系统的静态建模和行为建模。UML 静态模型包含对类及其属性、操作、接口的定义和类之间关联(例如继承、依赖和包含等)的定义。对系统行为语义的建模可以用序列图和协作图完成。CWM 规范主要使用了 UML 的静态图。

UML 语言由一个用 UML 表示的元模型(或语义模型)定义,这种递归定义使得整个 UML 可以基于非常少的(三个)未定义元素。此外,MOF 定义了 UML 的元元模型以表示一个递归的 UML 元模型的语义。

CWM 元模型直接从 UML 元模型继承而来,换言之,CWM 中的类都直接或间接地继承了 UML 中类的语法和语义。例如 CWM Relational 包中的关系模型定义了一个叫 Table 的类,表示任何关系数据库中的表,这个类继承于 UML 中的 Class 类。类似地,Column 类继承于 Attribute 类,这就建立了 Table 和 Column 之间的语义关系,即 Table 是一些 Column 的集合,这些 Column 具有一些共同的属性,但各自的属性值不同。这等同于 UML 中类和属性的关系,把 Table 和 Column 分别作为类和属性的子类就内在地确立了这种等同关系。CWM 这种直接从 UML 核心元模型中派生出数据仓库领域元模型的好处有很多,例如:

- CWM 元模型成为 UML 核心元模型的扩展,即它可以直接使用 UML 作为构建数据仓库领域元模型和模型(元模型的实例)的语言。
- CWM 可以直接使用 UML 图形标记表示数据仓库元模型。
- CWM 可以直接使用 UML 元模型中已经定义的语法和语义,无须重新定义,这通常指对抽象层次比较高的元类及其关联等概念的重用。例如,CWM Relational 包中

的元类 Table 和 Column 之间的关系继承了 UML 中的元类 Classifier 和 Feature 之间的关系，在 CWM 的 Relational 元模型中就无须明确地定义 Table 和 Column 之间的关联。

- CWM 元模型可以直接使用 UML 元模型中定义的数据类型。UML 元模型用元类 DataType 定义了数据类型的概念，在 CWM 中可以通过为 DataType 定义 M2 层描述的方式对其进行扩展，添加一些标准的数据类型，例如 SQL 或 CORBA 中的标准类型等。
- CWM 使用 UML 元模型作为描述面向对象型数据源的元模型。
- CWM 可以使用 UML 规范中定义的 OCL（Object Constraint Language，对象约束语言）表示对 CWM 元模型的约束条件。

2. MOF 与 CWM

MOF 是 OMG 用来定义元数据并将其表示为 CORBA 对象的一种技术，它支持任何能用对象建模技术表示的元数据，这些元数据可以按照用户需求在任意层次、程度描述系统的任何信息。

模型，作为一种对现实世界的描述方法也是元数据。模型的概念是高度可变的，它依赖于观察的角度。例如对于那些关心整个系统的人而言，模型应该包括系统所有的元数据。在 MOF 中，模型可以是任意具有抽象语法和语义的元数据的集合。元数据本身就是一种数据，因此它还可以被其他元数据描述。在 MOF 中，包含这种元数据的模型称为元模型。MOF 元模型定义了用 MOF 格式描述模型中元数据的抽象语法，因为一个系统中一般都会有很多类型的元数据，相应地就会有多种元模型。为了集成这些元模型，MOF 定义了一套通用的元模型构造语法，这种语法称为 MOF 模型，它是描述元模型的模型。实际上，它处于 M3 层，应该是元元模型，这里简称 MOF 模型。

MOF 规范由三部分组成：MOF 模型规范、MOF IDL 映射和 MOF 接口。

MOF 模型是 MOF 内部定义的元元模型，可以看作是定义 MOF 元模型的抽象语言，这和 UML 元模型定义 UML 模型类似，只不过前者是为了元数据建模，后者是为了对象建模。实际上，基于 MOF 的模型就是用 UML 标识的。MOF 主要提供四种组件构造 MOF 元模型，即类（Class），关联（Association）、包（Package）和数据类型（Data Type）。这些概念和 UML 中的类似，只是进行了一些简化，具体如下：

- 处于定义和实现层的类都可以拥有属性和方法。在 MOF 元模型中，属性表示元数据；方法提供对特定元模型中元数据的操作。属性和方法的参数可以定义为有序。
- 关联支持对类的实例的二元连接。每个关联都有两个端点，并可以对其排序性和唯一性等方面进行限制。如果一个类是一个关联的端点，那么它就可以包含通过这个关联到对方类的实例的引用。
- 包是相关的类和关联的集合。包之间可以引用、继承和嵌套。
- 数据类型使属性和参数可以使用非对象的类型。在 MOF 中，它们必须是能够用 CORBA IDL 表示的数据类型或接口类型。

MOF IDL 映射是一套将 MOF 元模型映射到 CORBA IDL 的标准模板。MOF 元素（M2 层）到 CORBA 对象（M1 层）的映射关系如下：类映射为元数据对象的 IDL 接口和元数据类代理，IDL 接口支持原类中定义的属性、操作和引用，而类代理提供对元数据对象的

代理操作。MOF 关联映射为元数据关联代理的接口,这个关联代理支持对关联的查询和更新操作。MOF 包映射为一个接口和元数据包代理,这个包代理实际上是包含原包中类和关联的代理的容器。

MOF 接口是表示 MOF 元模型的 CORBA 对象的接口,一般使用现有工具访问 MOF 模型库,其建模开发人员无须了解,只有那些开发基于 MOF 工具的程序员才需要了解。

OMG 已经采用 MOF 作为定义元模型的标准,CWM 元模型就遵循这个标准。这样,CWM 就可以使用 OMG 其他基于 MOF 的标准,尤其是可以用 XMI 交换 CWM 模型表示的数据仓库元数据,并用 IDL 和其他程序语言访问这些元数据。

3. XMI 与 CWM

XMI 的主要作用是用流的方式进行模型交换,因为 OMG 采用 MOF 表示元数据,XMI 的重点自然就是 MOF 元数据(即遵循 MOF 元模型的元数据)的交换。XMI 支持任何能用 MOF 规范表示的元数据的交换,它不仅可以对整个模型或部分模型组成的元数据进行编码,还可以对特定工具扩展的元数据编码。

XMI 可以看作是一种独立于中间件的通用的元数据交换格式,任何能够编写和解析 XML 数据流的元数据库或工具之间都可以进行元数据交换,它们无须实现 MOF 定义的 CORBA 接口,甚至根本不用支持 CORBA。XMI 还为非 MOF 的元数据库提供了一种交换方式,只要它能够将自身的元模型映射为 XMI 文档即可。

XMI 基于 W3C 的 XML,实际上就是一对并行的映射,一个在 MOF 元模型和 XML DTD(Document Type Definition,文档类型定义)之间,另一个在 MOF 元数据和 XML 文档之间。

XMI 主要由两部分组成,即:

- XML 文档生成规则——定义了将元数据编码为 XML 格式文档的规则,利用这个规则还可以将 XMI 文档解码得到元数据。
- XML DTD 生成规则——定义了为编码后的元数据生成 XML DTD 的规则,DTD 是 XML 文档的语法说明,一般的 XML 工具都可以用它来解析和验证 XMI 文档。

CWM 用 XMI 作为交换机制,这样数据仓库元数据和 CWM 元模型本身都可以充分利用 XMI 的灵活性和强大性。CWM 元模型通过 XMI DTD 生成规则产生一个标准的 DTD,而数据仓库元数据则可以通过 XMI 文档生成规则编码为一个 XML 文档。这大大增加了 CWM 的适用性,任何能够编写和解析 XML 数据流的元数据库或工具之间只需将自己的元数据转换为 CWM 模型的形式就可以利用 XMI 实现元数据交换,即它们所描述的数据交换。

1.5 数据粒度

数据仓库保存了大量的历史数据,为了保证数据的存储效率和组织清晰,数据仓库的数据以不同粒度进行存储。

数据仓库中存在不同的综合级别,一般称为“粒度”。粒度越大,表示细节程度越低,综合程度越高。四种粒度级别分别是早期细节级、当前细节级、轻度综合级和高度综合级,分别反映不同的数据需求。由此可知,数据仓库中数据生存周期,即源数据经过综合后首先进

入当前细节级，并根据具体需要进一步的综合，从而进入轻度综合级乃至高度综合级，老化的数据将进入早期细节级。数据仓库的核心是在系统中保留最有可能被用户使用的数据。

1.6 数据模型

数据模型是对现实世界的一种抽象，根据抽象程度的不同，形成了不同抽象层次上的数据模型。与关系型数据库的数据模型类似，数据仓库的数据模型也分为如下三个层次。

1. 概念模型

概念模型是客观世界到计算机系统的一个中间层次，最常用的表示方法是E-R(Entity-Relationship，实体-关系)图。目前，数据仓库一般建立在数据库基础上，所以其概念模型与一般关系型数据库采用的概念模型一致。

2. 逻辑模型

逻辑模型是数据的逻辑结构，如多维模型、关系模型和层次模型等。数据仓库的逻辑模型描述了数据仓库主题的逻辑实现，即每个主题对应的模式定义。

3. 物理模型

物理模型是逻辑模型的具体实现，如物理存取方式、数据存储结构、数据存放位置以及存储分配等。在设计数据仓库的物理模型时，需要考虑一些提高性能的技术，如表分区和建立索引等。

目前，对数据仓库数据模型的讨论大多集中在逻辑模型，其中最常用的是多维模型。在多维模型中，涉及以下一些基本概念。

维是指人们观察数据的特定角度。例如，企业常常关心不同销售数据随时间的变化情况，所以时间就是一个维度。

维度层次是指人们观察数据的某个特定角度还可以存在细节程度不同的多个描述，即维度层次。一个维度往往有多个层次。例如描述时间维度时，可以有年、季度、月和日等不同层次，则年、季度、月和日就是时间维度的层次。

维成员是指维的一个取值。如果一个维是多层次的，则该维度的成员就是在不同层次上取值的组合。例如时间维有年、月和日三个层次，则分别在三个层次上各取一个值组合起来就得到时间维度的一个成员，即“某年某月某日”。

度量描述了要分析的数值，例如销售额等。

粒度是指数据仓库所保存数据的细化或综合程度的级别。细化程度越高，粒度越小；反之，细化程度越低，粒度越大。

1.7 ETL过程

业务系统的数据经过抽取、转换并加载到数据仓库所在的中心存储库的过程称为ETL(Extract, Transform and Load)过程，制定这一过程的策略称为ETL策略，而完成ETL过程的工具则是ETL工具。相对于数据仓库的表而言，业务系统数据库中的表称

为源表,业务系统数据库称为源数据库,数据仓库中所有的数据都来自于业务系统数据库。在构建数据仓库的过程中,ETL 的实施是一项烦琐、冗长而艰巨的任务,因为它关系到数据仓库的数据质量问题。如果导入的数据漏洞百出,那么对决策者而言无疑会造成困扰。

1.7.1 主要流程

随着应用和系统环境的不同,数据的抽取、转换和加载具有不同的特点。一般地,ETL 主要流程如图 1.10 所示。

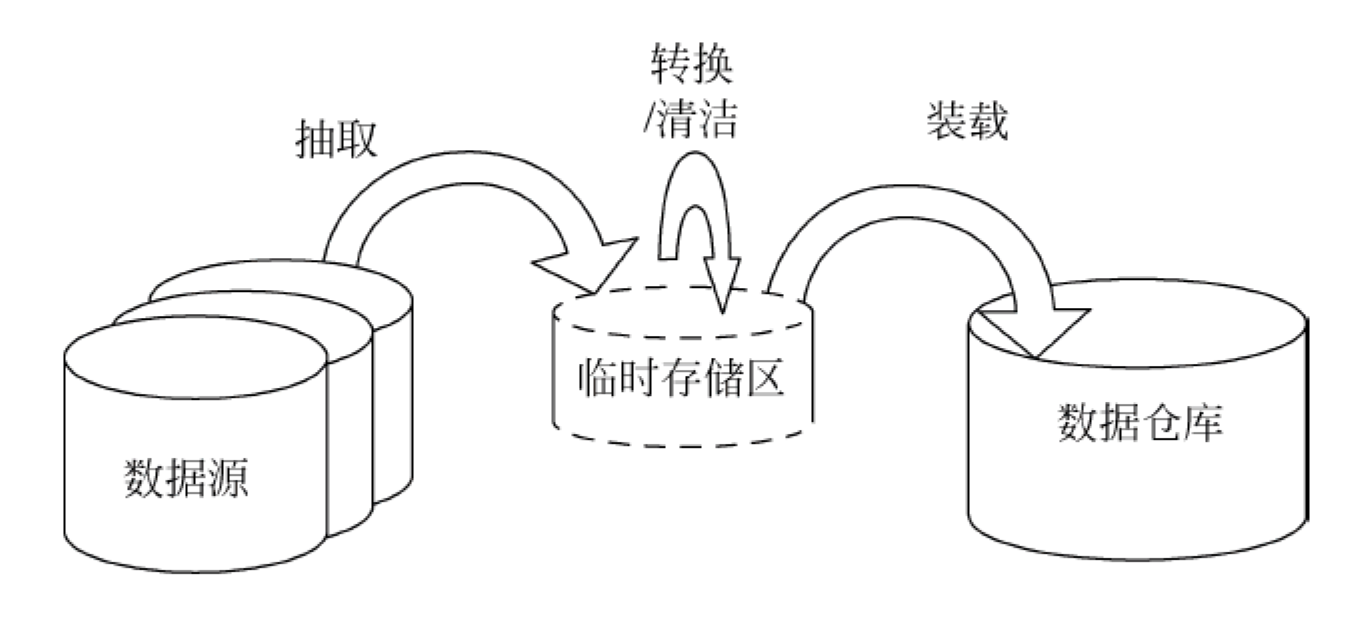


图 1.10 ETL 流程

(1) 预处理是正式开始作业之前的准备工作,包括清空工作区、检查过渡/准备区。在需要直接访问操作型数据源系统时,应检查远程数据库服务器状态,并核对目标区数据加载状态,以核算出加载作业的参数,如加载数据的时间间隔和范围(是 24 小时的数据,还是前 3 天的数据)。

(2) 启动数据加载的批作业。

(3) 因为维表有事实表所参照的主键,所以需要先完成对维表的加载,生成维表主键,并作为以后加载事实表所需的外键。在加载维表时,有时需要处理好缓慢变化的维,并可能涉及版本号的处理问题。

(4) 加载事实表。这中间也涉及键查找的问题,即从有关维表中找到相应的主键,并以此作为事实表的外键。

(5) 事实表加载完成后,再对实体化方阵体系进行刷新,以保障实体化方阵与其基础数据同步。

(6) 设计具有完善的出错处理机制和作业控制日志系统,以监测和协调整个加载过程。

具体的 ETL 处理流程如图 1.11 所示。

1.7.2 数据抽取

数据抽取是 ETL 处理的首要任务,解决的主要问题是确定需要抽取的数据,并采用适当的方式抽取。

源数据进入数据仓库是通过数据抽取完成的,从一个或多个源数据库中通过记录选取

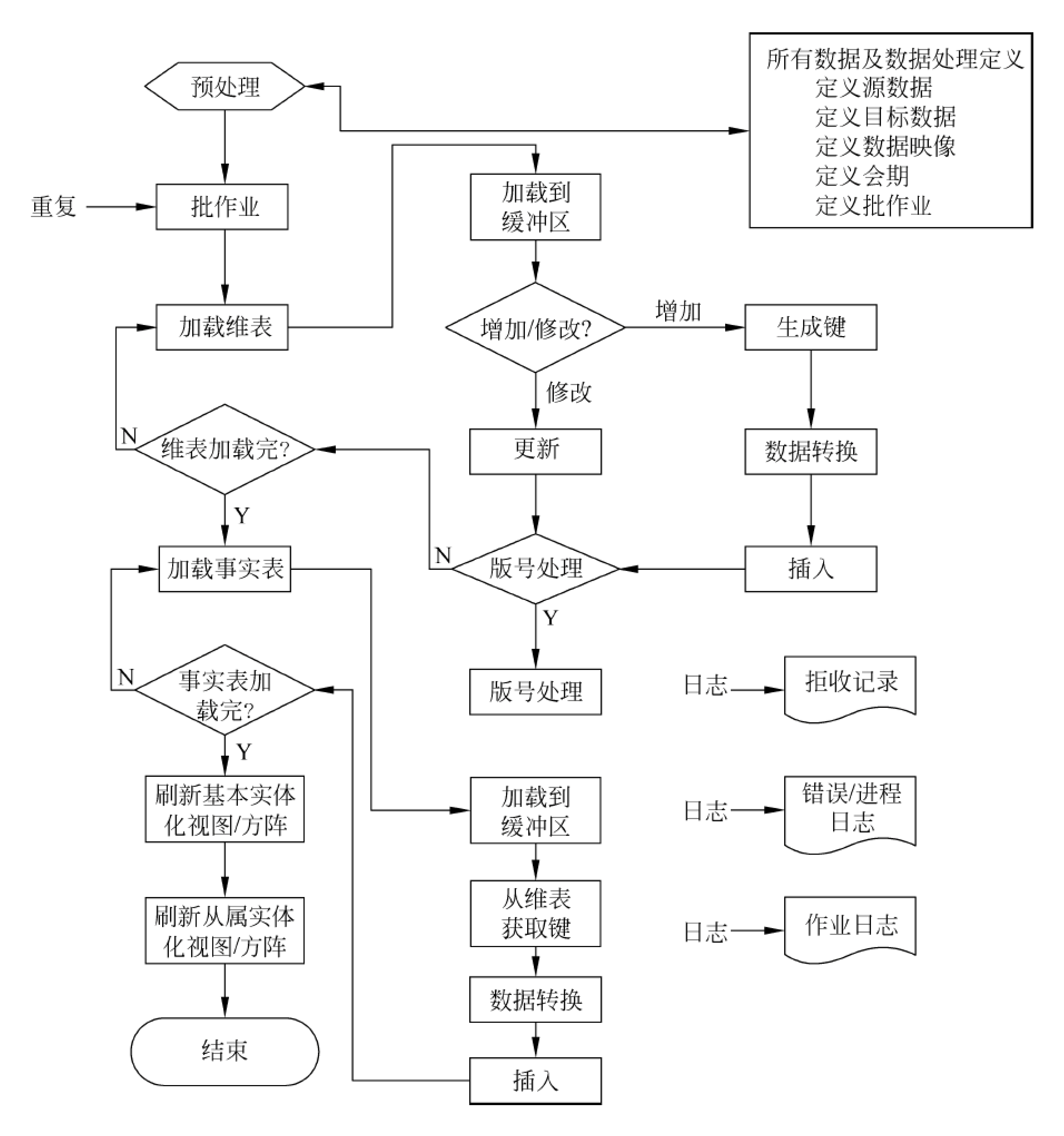


图 1.11　ETL 处理流程

进行数据复制的过程。抽取过程将记录写入 ODS(Operational Data Store,操作数据存储)或者临时存储区(Staging Area)以备进一步处理。

数据抽取的主要功能包括：

- 数据提取——主要是确定要导入数据仓库的数据。
- 数据清洁——检查数据源中存在矛盾的数据,按照用户确认的清洁规则对数据进行修改。
- 数据转换——主要是将数据源的数据转换成数据仓库要求的格式,其中包括数据格式的转换,例如将数据源中的日期字段转换成数据仓库要求的字符形式;数据内容的转换主要是将同一含义的字段用统一的形式表达;数据模式的转换是由于数据仓库系统和业务系统面向的数据操作不同,所以在数据模式上也存在不同,例如电信业务的出账表的主键包括用户标识和费用项,但是在数据仓库中用户主题中用户账务信息采用用户标识作主键,将不同费用项的费用作为字段,这样就需要在数据抽取时进行不同数据模式的转换。

- 生成衍生数据——由于数据仓库保存了大量的历史数据同时需保证查询的效率,需要对用户经常进行的查询通过预处理操作来提高查询效率,生成衍生数据。衍生数据既包括某些数值数据的预运算,例如平均值和汇总等,也包括某些分类字段的生成,例如对用户费用的分档信息等。

数据抽取的重要组成部分是变化数据捕获(Change Data Capture,CDC)。实现CDC的方法包括时间戳、读取RDBMS系统的日志文件、使用源系统中的触发器或者自行开发CDC程序来检查日志文件等。时间戳是最简单的,也是相当普遍的方法。前提是所有的源表都有时间戳。例如超市的业务系统数据库中几乎每个表都有“插入记录日期”和“更新记录日期”两个字段,这是规范建库的一个基本标准。

在多数情况下,数据源与数据仓库并不处于同一个数据服务器中,它们往往是独立的,并处在远程系统中。数据抽取往往是以远程、分布式的方式进行,并涉及各种各样的方法和手段,其中包括:

- 应用SQL * Plus提取到文本文件。
- 应用OCI或Pro * C程序,或者Oracle UTIL_FILE提取到文本文件。
- 应用Oracle Export Utility程序提取到Oracle Export Files文件。
- 远程数据复制。
- 信息流。

如果来自操作型数据源的数据含有不洁的成分和不规范的格式,将对数据仓库的构建和维护,特别是对联机分析处理带来很多问题和麻烦。因此,必须在ETL中加以解决,通常包括以下的处理方法:

- 拼写检查、分类并与标准值对照检查。
- 处理名字和地址。
- 为名字和地址建立辅助表或联机字典,据此进行检查修正。
- 数据转换函数以及子程序库。

保障数据清洁顺利进行的原则是优先对数据清洁流程进行分析和系统化的设计,针对数据的主要问题和特征,设计一系列数据对照表和数据清洁程序库的有效组合,以便应对不断变化的、形形色色的数据清洁问题。

数据清洁通常包括:

- 预处理——对于新的数据加载文件(特别是新的文件和数据集)要进行预先诊断和检测,不能贸然加载。有时需要临时编写判断小程序,称作“小狗”,它会用鼻子闻一闻这个庞然大物(不知底细的文件),以进行“卫生检查”。
- 标准化处理——应用数据仓库内部的标准字典,对地名、人名、公司名、产品名、品类名进行标准化处理。
- 查重——应用各种数据查询手段,避免引入重复数据。
- 出错处理和修正——将出错的记录和数据写入日志文件,留待进一步处理。

数据仓库必须存放“优质数据”,即符合一致性的、大家公认或经过验证是有价值的,并符合元数据定义的。通过数据清洁能够检测出那些违反规则的数据,一旦发现要么抛弃,要么将其转换成“清洁”数据,使其符合规则,然后再装载到数据仓库。

1.7.3 数据转换

数据转换是将抽取出的数据进行过滤、合并、解码和翻译等,为数据仓库创建有效数据的过程。一旦数据抽取完成,就需要设计并确定转换规则应用于已抽取出的数据。数据转换需要理解业务侧重点(Business Focus)、信息需求(Informational Needs)和目前可用的数据源。

常用的转换规则包括:

- 字段级的转换,主要是指数据类型转换;增加"上下文"数据,如时间戳;解码(Decoding)即将数值型的地域编码替换成地域名称等。
- 清洁和净化,主要是保留字段具有特定值或特定范围的记录;引用完整性检查;去除重复记录等。
- 多数据源整合,主要是字段映射(Mapping);代码变换(Transposing)即将不同数据源中的数据值标准化为数据仓库数据值。例如,将原系统非英文编码转换成数据仓库英文编码;将原系统信息编码转换成数据仓库信息编码等;合并(Merging)即将两个或更多源系统记录合并为一个输出或"目标"记录;派生(Derivation)即根据源数据,利用数学公式产生数据仓库需要的数据。例如,由身份证号码计算出出生日期、性别、年龄等。
- 聚合(Aggregation)和汇总(Summarization)。

虽然,数据转换较为烦琐,但却是 ETL 三个步骤中最简单的。许多 ETL 工具都提供了强大的转换功能,例如 DTS 中有复制字段转换、小写字符串转换、大写字符串转换、中间字符串转换、剪裁字符串转换、日期/时间字符串转换、读取文件转换、写入文件转换和 ActiveX 脚本转换等,其中最常用的是 ActiveX 脚本转换,因为它允许自行撰写 VBScript 或 JScript 将原始字段中的数据转换至目标字段中的数据。事实上,如果上述的各种转换类型都无法满足需求时,则 ActiveX 脚本转换将是最终的选择。

1.7.4 数据加载

数据加载是将转换/清洁后的数据装载到数据仓库,实现数据加载可选用的实用程序和工具很多,例如最基本的 Import、SQL Loader 和 SQL 语言等。为了提高程序和过程的复用性,编写和设计数据加载的函数库/子程序库是十分必要的。

数据加载包括维表和事实表的加载,两者具有不同的加载策略。

1. 维表加载策略

本质上,有三种维表加载策略。每种策略按不同的方式处理维表中的变化以及更新维表数据或捕获属性的变化历史。这三种加载策略统称为慢速变化维表策略(Slowly Changing Dimension,SCD)。在这三种策略中,所有输入数据都与现存的数据进行比较,如果在自然键上没有发现匹配的记录,那么输入的记录就被插入维表中。自然键由维表中的一些列(数据项)组成,能唯一确定维表中一条记录的代理键不包括在这些列中。

1) 慢速变化维表类型 1(SCD-1)

在 SCD-1 策略中不需要保存历史记录。如果一条输入记录已经在目标维表中(根据自

然键值进行判断),则可根据输入记录的数据对该记录进行更新或刷新。

2)慢速变化维表类型2(SCD-2)

有时记录中有一些重要值(即维表中一个或多个列组成的自然键值)需要保留。在SCD-2策略中,当相关事实发生时,就可保存一条维表记录。因此,如果一条输入记录中的某个字段或列值属于"重要值",并且它与目标表中相应的列值不同,则现存的记录就已经"过期"了,则需要根据输入的记录,在维表中插入一条新记录并指派一个新的代理键值。如果输入记录中没有一个重要值与目标表中相应的列有所差别,则对现存记录进行更新,而不是使它过期。

3)慢速变化维表类型3(SCD-3)

SCD-3与SCD-2非常相似,用于跟踪重要值的变化,然而这种策略不是为每个变化添加一条不同的记录,而是在现存的记录上使用不同的列来保存当前值和任意的前n个值。当探查到一个重要值发生变化时,该字段所有以前的列需要向下一列移动,第n个以前的值将被丢弃。但是这种策略不被很多数据仓库专家看好,所以不建议使用,因为这将加大ETL的难度。

所有的维表加载都遵循上述三种策略,但有两个维表例外,即静态维表和完全由新数据替换的维表。静态维表不会变化,如果它发生变化,则发生的唯一变化是在其中添加一条新记录,时间维表就是一个静态维表;完全由新数据替换的维表通常是小的代码列表,它们不需要代理健或变化历史。但需要注意的是表中任何重要值的丢失都是不可承受的。

2. 事实表加载策略

事实表的加载是必须的也是非常重要的,它是后续数据分析的基础。事实表的加载就是持续不断地增加数据。事实表的加载不是简单的数据复制,必须首先将每个事实表与各自的维表的代理键相结合,每个源表(与事实表关联的表)必须有足够的信息用于查找维表中的自然键,以验证事实记录的完整性。

在数据加载过程中,经常涉及主键查找的问题。主要是对某些键查找函数程序进行修改补充,原因是这些键查找程序需要异常控制(Exception Handling)以返回两种不同的结果(找到的键值或返回空值)。

加载维表的过程中,在插入新的记录或修改已有记录之前,需要比较某些逻辑键来确定当前记录是否存在。在加载事实表的过程中,经常涉及更多的主键查找处理。我们知道,数据仓库的事实表含有许多外键,并以此与有关维表的主键相关联。在进行事实表加载时,往往需要查找有关维表的主键值,并以此确定事实表的外键值。常用的方法是,通过从源数据提取的数据部分的逻辑键与数据仓库内有关维表的逻辑键进行比较和匹配,如果匹配,则取维表的主键值,并以此作为事实表的外键值加载到事实表中。例如对客户维度而言,可以通过客户的姓名、邮政编码与客户维度进行比较,如果匹配,则以客户标志符键作为事实表的外键。

逻辑上,实现ETL主要采用三种策略,即远程抽取(Remote Extraction)、推(Push)和拉(Pull)。其中,远程抽取是指ETL过程是在一个独立的远程平台上进行,这种方式受数据源现有系统或目标数据仓库的结构影响最小,几乎不影响源和目标系统的CPU和容量,但可能会提高成本,对网络带宽要求较高。推是指在现有系统环境中开展ETL过程。当现有系统具有相当充裕的容量,而且不会超出CPU处理限度时,经常采用这种方法。当

现有系统是同构的，也经常采用这种方式。拉是指 ETL 过程在“仓库”端进行，当现有系统是异构时，经常采用这种方式。当“仓库”端平台具备必需的性能和容量时，将体现出一定优势。但是由于“仓库”的不断增长，可能需要制定长期策略。

1.8 数据质量

对于企业而言，数据已经成为企业战略和在激烈的市场竞争中取得优势地位的关键。企业管理的核心是决策，而决策的依据是数据，只有准确把握数据和信息，才能为企业的运营把脉。深刻分析数据质量管理的主要目标，明确数据的分类与分布，以进一步提高企业数据质量，保障数据挖掘技术的成功应用和有效实施。

1.8.1 主要问题

随着应用系统数据量的急剧增加、新应用的不断出现以及应用之间的相互整合，数据质量问题变得日益突出。目前企业存在多地区、多产品、多业务和多系统的环境，在数据信息、报表统计、业务分析和业务定义方面存在数据不一致、不完整、不统一、不真实和不及时的情况。导致业务人员、管理者、信息使用者对数据信息存在误解。质量低劣的数据已成为影响企业正确决策的一个重要因素。出现这种现象的主要原因可归纳为如下几个方面。

1. 多系统分散建设

为适应迅速变化的市场和竞争环境，企业构建了各种面向特定应用或部门的系统。系统的分散建设导致同一类数据分散在不同的应用系统中，缺乏统一的数据来源和技术标准，导致出现数据不规范、不一致和无法共享等问题。多系统提供数据，缺乏企业级统一、可信的数据源。例如客户数据分布在多个核心业务系统中，存在一个客户对应多个用户 ID 的情况。

2. 数据多头管理

由于系统的建设和管理职能分散在各部门，而各部门中数据管理的职责分散，权责不明确，致使数据管理的相关规范无法有效执行和落实。在企业层面，缺乏统一的数据管理组织，致使企业无法统一定义数据管理的规程和标准等，相应的数据管理监督措施无法得到落实，企业范围内的数据考核体系还尚不完善，无法保证数据管理标准和规程的有效执行。同时，也缺乏对于数据管理的组织和岗位职责的界定，各部门数据管理人员安排自行决定。如销售人员管理系统中对各个机构名称不规范。信息部门无权修改各家分公司中机构的名称，而业务部门往往意识不到规范名称的重要性。

总之，目前企业缺乏统一的数据管理组织，数据管理职责分散，导致权责不明，存在“人人管理，无人负责”的现象，缺乏统一完善的数据管理制度和规范，以及落实这些制度与规范的监督考核机制。

3. 缺乏统一标准

由于系统分散在各部门进行管理或使用，目前尚没有建立企业级统一的数据标准。因此当前的数据标准大部分由各部门分别制定，从而导致数据管理过程中缺乏统一规范，具体体现在以下几个方面：

- 尚未建立企业级统一的数据编码，各业务部门根据自身的业务需要和系统要求，建立自己的数据编码，从而导致各类数据编码不一致，甚至发生冲突。
- 数据模型主要是在系统的建设过程中由各部门根据需要分别制定，导致数据模型不统一，甚至部分数据模型存在缺陷。
- 数据指标体系主要是各业务部门根据系统建设的需求分别制定，对于指标的口径解释不一致，且部分数据指标缺失。

总之，目前存在不同业务在不同系统中对同一数据指标的解释和编码方式不一致，造成数据口径不统一，数据模型也自成体系，导致没有企业统一的数据模型。

4. 缺乏完善的数据质量管理体系

目前，企业的数据质量管理主要由各专业部门分头进行，数据稽核措施停留在专业部门内部，部门间缺乏交流，各自为政；跨部门的数据质量沟通机制不完善；缺乏清晰的跨部门的数据质量管理规范和标准，数据分析随机性强，存在业务需求不清的现象，影响数据质量；尚未全面实现数据的自动采集，处理过程存在人为干预。同时，各部门也存在数据质量管理人员不足、知识与经验不够、监管方式不全面等多方面的问题；缺乏完善的数据质量管控流程和系统支撑能力。

5. 数据生命周期管理不完整

缺乏完善和统一的数据生命周期管理规范和流程，不能确定过期和无效数据的识别条件，且非结构化数据未纳入数据生命周期的管理范畴；无 IT 工具支撑数据生命周期状态的查询，未有效利用元数据管理，数据生命周期的管理水平有待提升。

6. 对数据安全认识不全面

目前，企业对数据安全的认识主要集中在数据安全策略和用户授权访问方面，对数据安全事件处理和数据风险管理等方面认识不够全面。各部门需要设置数据安全管理岗位，明确职责。对数据安全级别的定义不统一，信息披露审批机制有待完善。

1.8.2 评价标准

数据质量可以从数据的完整性、唯一性、精准性、合法性、一致性、及时性和关联性七个方面进行评价，如图 1.12 所示。

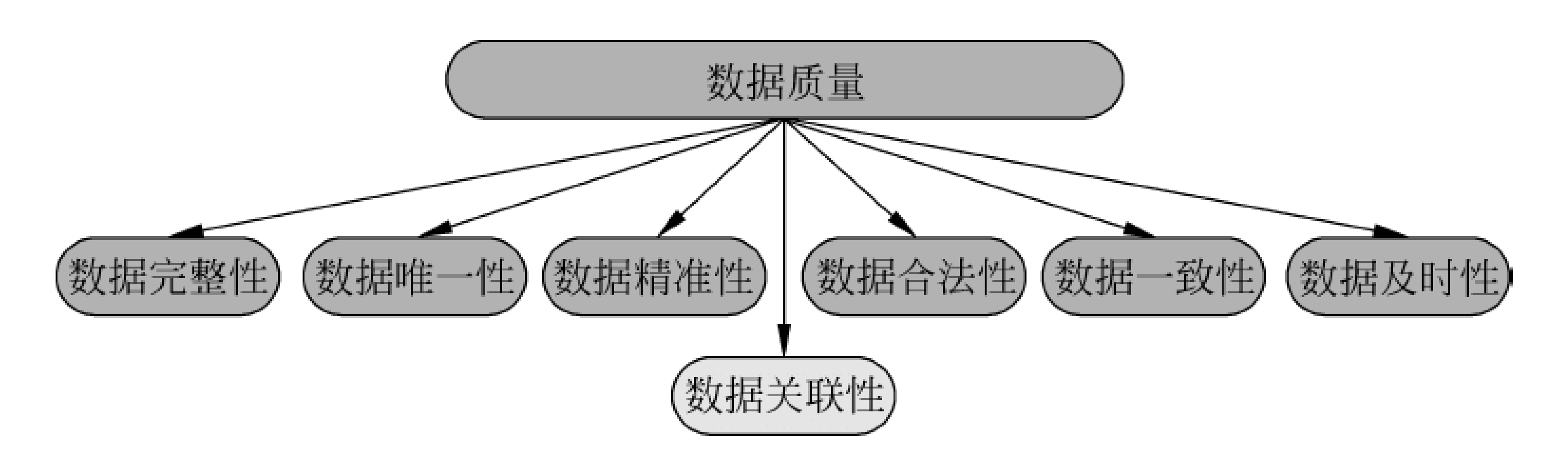


图 1.12 数据质量评价标准

1. 数据完整性

数据仓库的建设过程是将不同源系统的数据逐步集成到仓库，随着系统建设的推进，数据越来越丰富，数据仓库支持的功能越多，数据的完整性越来越强。换言之，数据

完整性是指实体不缺失、属性不缺失、记录不缺失和字段值不缺失四个方面。数据是否完备，是衡量数据质量的一个重要指标。数据源数量随时间的变化情况如图 1.13 所示。

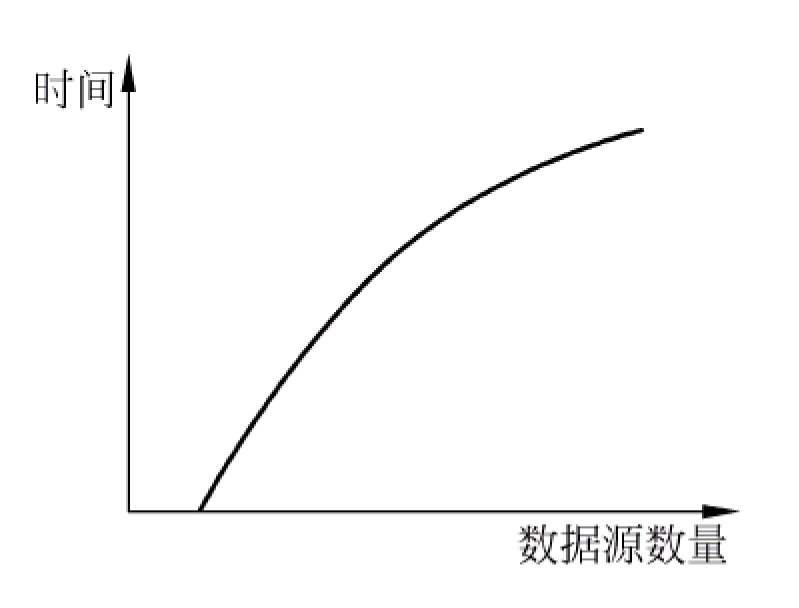


图 1.13 数据源数量随时间变化的示意图

2. 数据唯一性

数据唯一性表现在两个方面：一方面同一数据的来源唯一；另一方面，主键唯一和候选键唯一。就数据来源唯一而言，例如就截至某年某月的用户数来说，有两个系统数据来源 HLR 和 BOSS，并且这两个用户数还不尽相同。这就要求根据数据的用途选择数据的来源。

3. 数据精准性

数据精准性包含两方面的含义，即准确和精确。准确是指从数据仓库系统展示给用户的从不同角度组合出来的不同指标值的真实程度，一般是将业务系统的值跟仓库展示出来的值进行比较，这是数据质量最复杂、最难于控制的一个指标；精确是指计量误差和度量单位等方面的精确程度。

4. 数据合法性

数据合法性包含两方面的含义：一方面是指数据来源的合法性，例如某地某年的 GDP 数据、某年某月公司的利润数据，都是经权威部门发布的正式渠道为来源，并在数据属性中标明来源；另一方面是指数据操作过程中格式、类型、阈值和业务规则的有效性。

5. 数据一致性

数据一致性是指数据仓库中的数据与源系统的同一数据在不同层面、不同人员的一致理解。在系统内，又表现为在不同的处理环节，信息的表示一致、含义一致、展现一致和口径一致。

6. 数据及时性

数据及时性是指在 OLTP 系统中变化或新产生的数据，需要经过多长时间才能到达数据仓库。数据使用是有时限要求的，因此数据刷新、修改和提取等操作的及时性和快速性是一个重要因素。通常，引入数据延迟时间(Data Delay)和数据访问响应速度加以衡量。其中，数据延迟时间是指业务系统发生变化或新增数据，体现到仓库数据所需要的时间。

7. 数据关联性

数据关联性是指源系统不同数据之间存在的内在业务逻辑关系承袭到目标系统数据中，并使这种业务逻辑在目标系统内部数据间保持不变。例如计费系统中的通话时长和计费时长，在分析每爱尔兰收益时，这两个数据的逻辑关系就表现得非常重要。通常，用源系统的不同数据之间的关联来验证目标系统数据间的关联。

1.8.3 管理目标

企业数据管理经历了初始关注数据生产，进而关注数据管理，最终关注数据服务三个发展阶段，如图 1.14 所示。

- 数据生产阶段：数据生命周期(数据的产生、整合、存储和应用)和元数据。

• 数据管理阶段：基于元数据，对数据生命周期的管理，强调对数据标准、数据所有权、数据安全和数据质量的管理。
• 数据服务阶段：在数据管理基础上，关注数据服务的需求和使用。

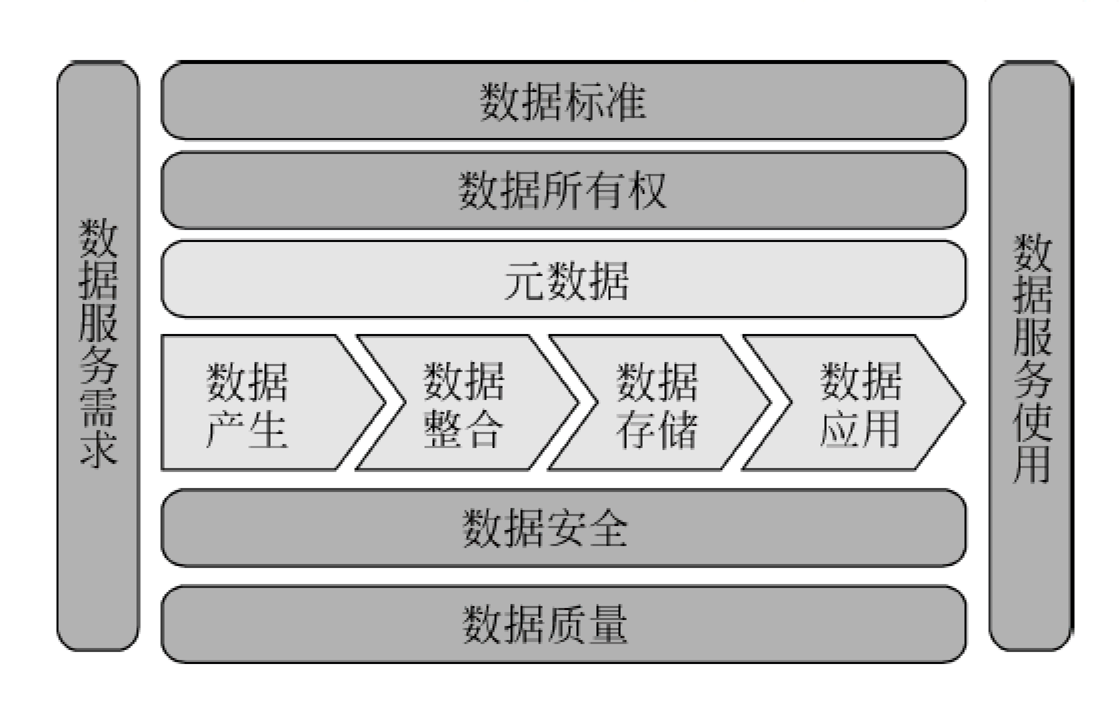


图 1.14 数据管理的三个阶段

针对企业目前数据管理方面存在的问题，需要制定数据管理目标，即实现数据整个生命周期的统一管理，实现统一数据管控，提供全面、统一的数据服务，灵活支撑业务，为企业精细化管理提供保障，为企业发展创造价值，以提升企业运营与管理能力，打造企业核心竞争力。

因此数据管理需要达到以下六个方面的要求，即：

• 听得懂。统一数据“语言”，要求数据管理“有标准”，即具备完善的数据管理标准规范，保证在企业内遵循并应用统一的数据标准。
• 不出错。在数据整个生命周期的各个环节建立完善的数据质量稽核机制，确保数据的准确性，要求企业具备完善的数据质量管理规范。
• 不乱改。明确数据的所有权和更改权限，制定完善的数据所有权管理规范，确保对数据的所有更改均有“法”可依，有“据”可查；考虑是否可以采用增量修改技术。
• 不丢失。建立数据的备份和容灾机制，制定完善的数据安全管理规范，确保所有数据备份，可恢复。
• 不泄露。根据内容和企业信息保密委员会的相关要求做好数据保密工作，防止信息泄露。
• 易使用。制定完善的数据服务管理规范，保证数据易获取、易应用，以充分发挥数据作为企业资产的价值。

1.8.4 管理体系

为了实现数据管理目标，构建完整的数据管理体系，必须从制度、流程和操作、数据清理和优化三个方面建立数据管理体系。明确的数据管理制度定义了数据管理应当遵循的规范，可以有效地保障数据流程及操作的落实；完善的流程是在统一数据标准的基础上，进一步提高数据质量、保证数据安全所必须遵循的行为规程；数据清理和系统优化为数据规范管理提供支撑。这三方面构成了完整的数据管理体系，保证数据管理有标准、有制度、有稽核、有手续、有手段、有检查、有考核，使得各项数据管理工作能够得以有效落实，达到数据准

确、完整的目标，并能提供有效的增值服务。

1. 数据管理制度

根据数据管理的目标和主要内容，明确数据管理具体角色和职责的划分。根据企业的组织架构以及各部门的职能划分，数据管理涉及数据管理部门（如信息技术部）、业务销售部门（如业务管理部、个人业务部、集团业务部等）、数据运营部门（如系统开发和运营部门）以及数据服务部门（如客户服务部等），如图 1.15 所示。

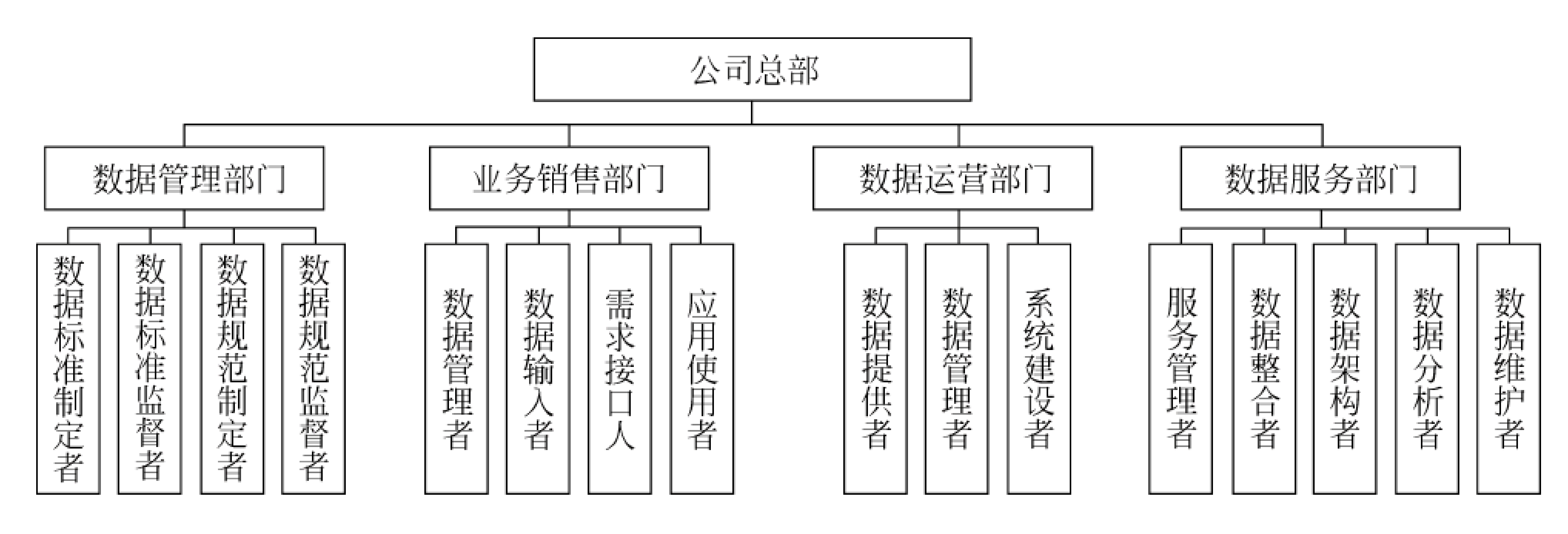


图 1.15　数据管理的职责划分

2. 数据管理标准

没有标准就无法界定数据质量的好坏，因此有必要制定数据标准。数据标准的制定需要业务部门、销售部门人员的大力配合。初期先制定重要数据（如客户、产品等）的标准，之后不断丰富和完善。数据标准是系统建设的依据，按照统一的数据标准进行系统建设是保证数据准确性和一致性的基础。在数据管理方面，企业可考虑制定如下的企业内部标准：

- 数据编码——数据编码体现在企业业务流程的各个环节，它定义了企业所有管理对象的唯一编码，其目的是实现企业业务流程的标准化。
- 企业数据模型——描述企业所有核心数据及其相互关系的统一全局视图，是所有系统建设必须遵守的重要依据。
- 企业数据指标体系——统一企业数据的指标解释，保证数据的一致性和准确性。

明确系统主数据一方面利于界定权威系统从而制定合理的数据整合策略；另一方面利于 IT 系统建设的各项目组统筹考虑数据的采集和访问。

3. 数据管理流程

数据管理流程是实施规程以规范人和系统在数据质量、安全等方面的管理行为，是在统一数据标准的基础上，进一步提高数据质量、保证数据安全所必须遵循的行为规程。如图 1.16 所示为一个完整的数据管理规程体系，这些管理规程涵盖了数据管理的全部工作内容，是做好数据管理的依据。

4. 数据服务管理模型

企业数据服务管理需要从组织、流程、规范和技术四个角度开展全方位的企业数据服务管理工作，实现数据生产、数据管理和数据服务的全面提升。下面分别阐述数据服务管理模型中的组织视图、流程视图、规范视图和技术视图，分别如图 1.17～图 1.20 所示。

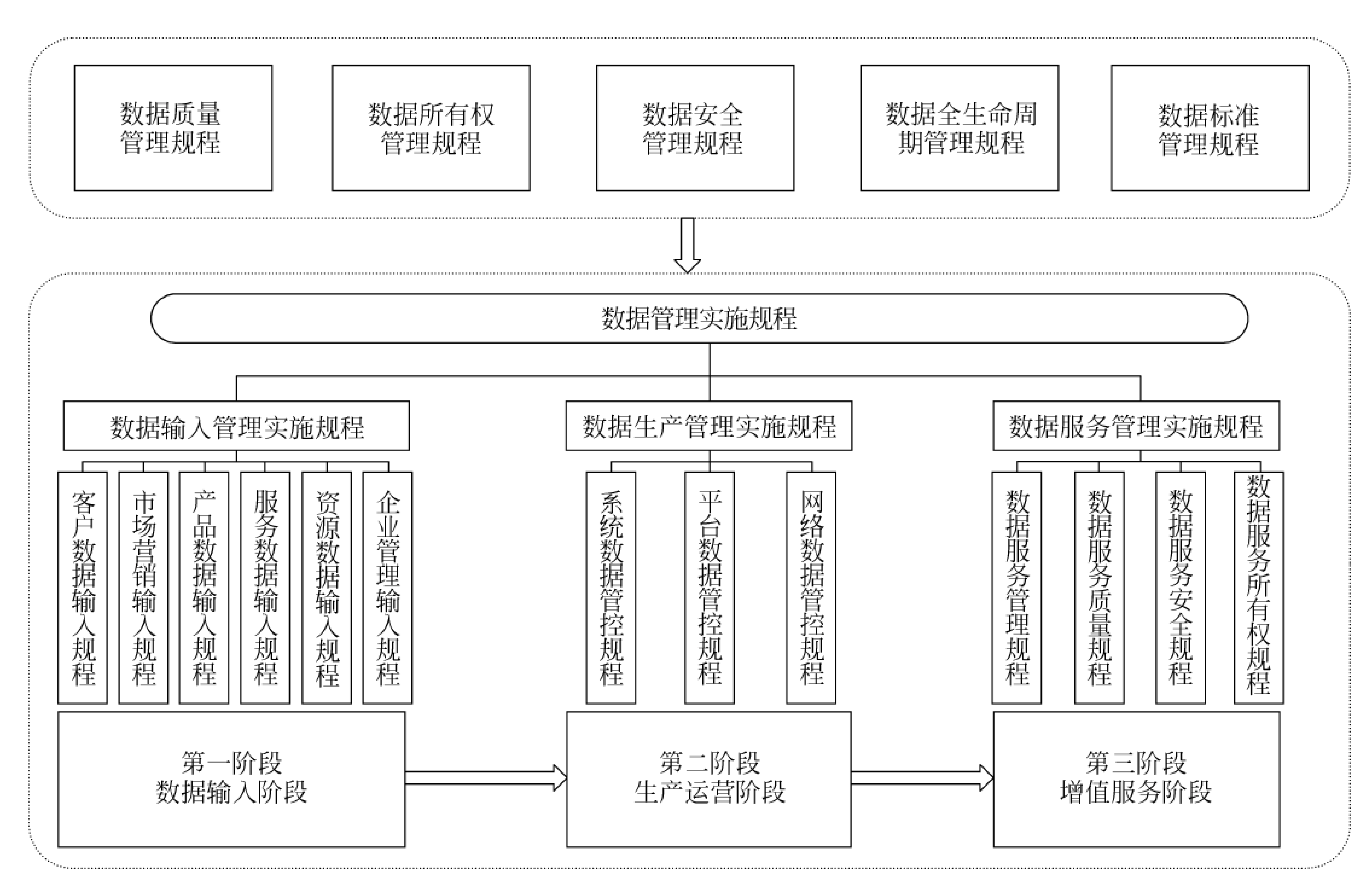


图 1.16　数据管理规程体系

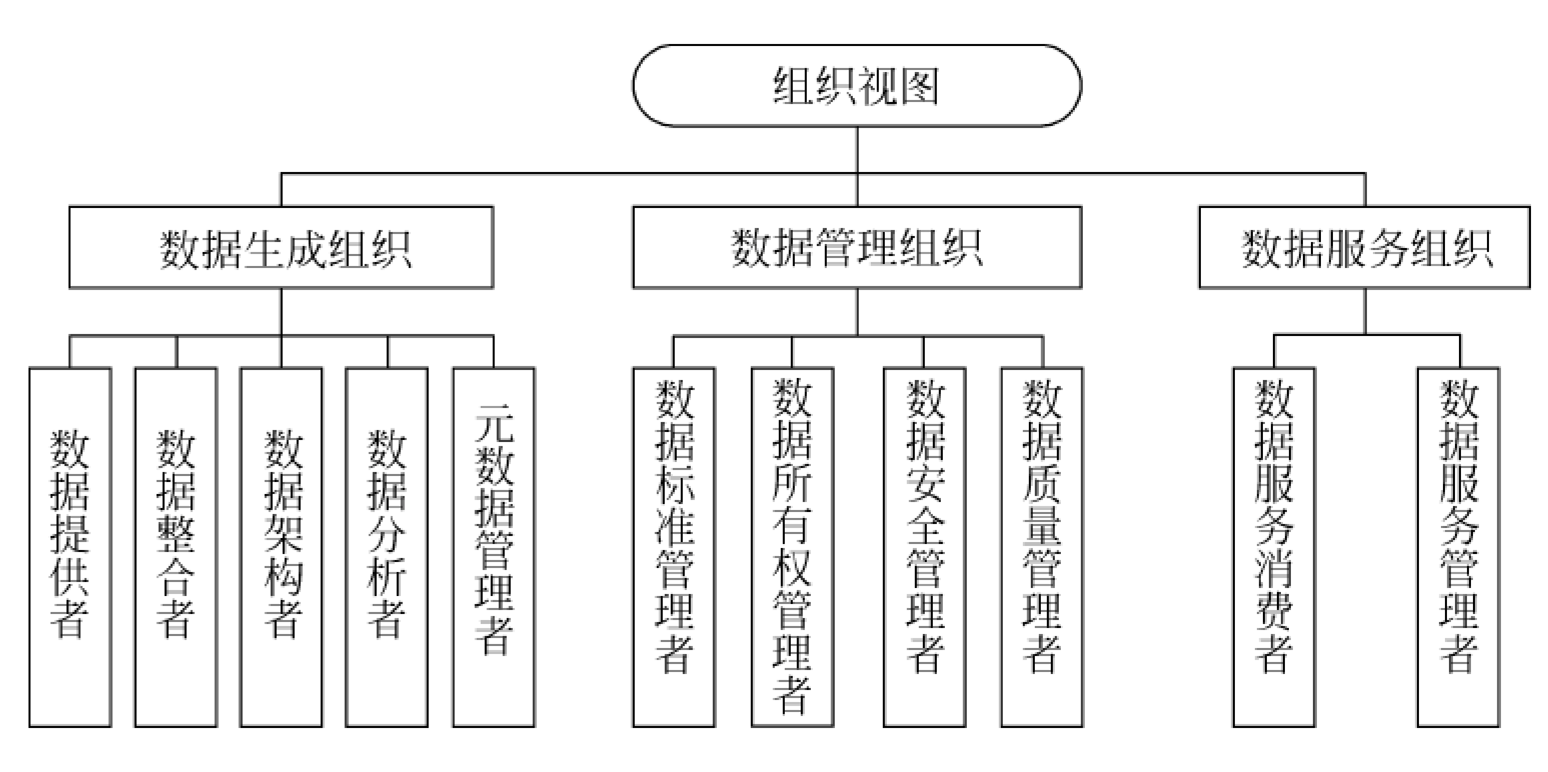


图 1.17　数据服务管理模型的组织视图

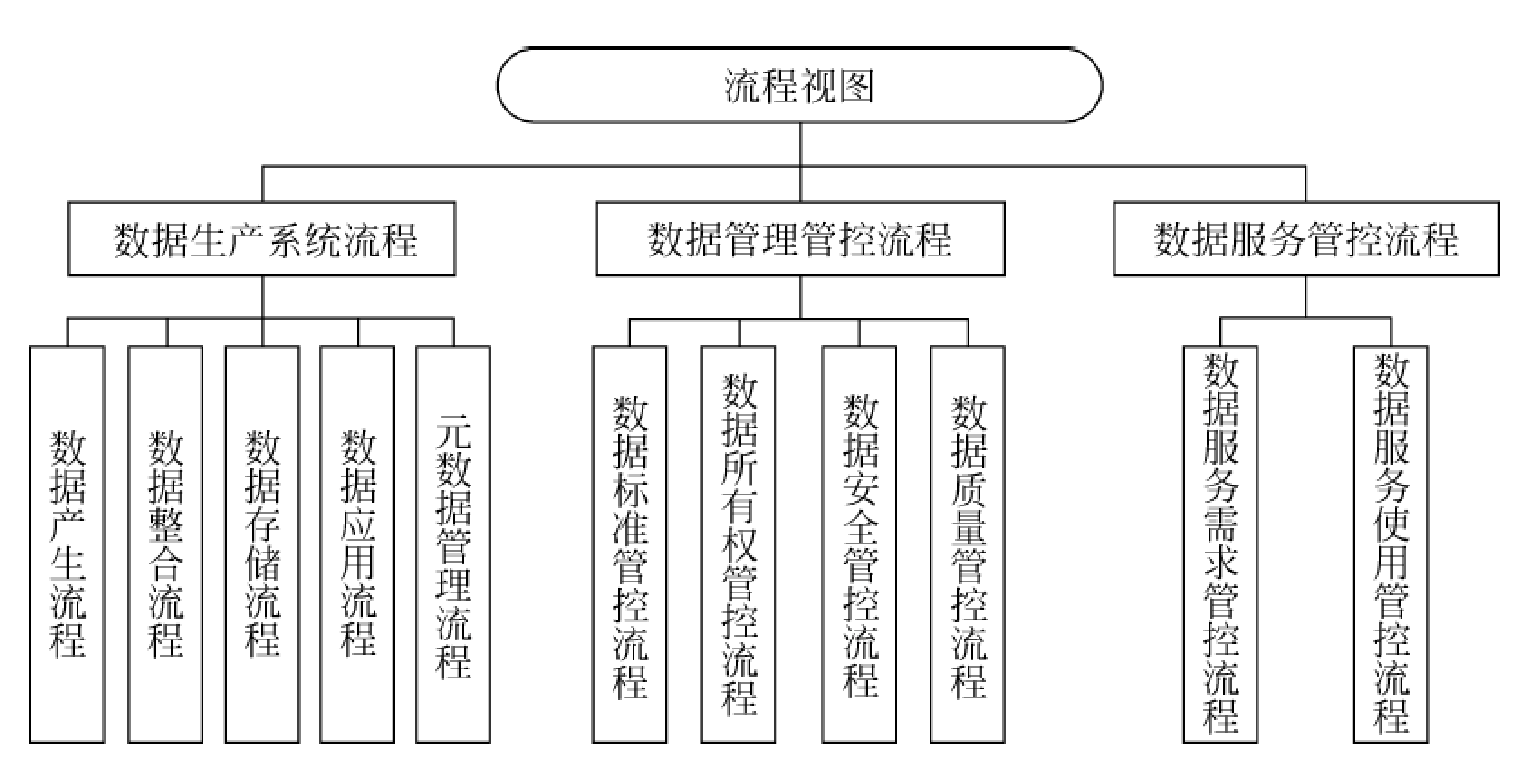


图 1.18　数据服务管理模型的流程视图

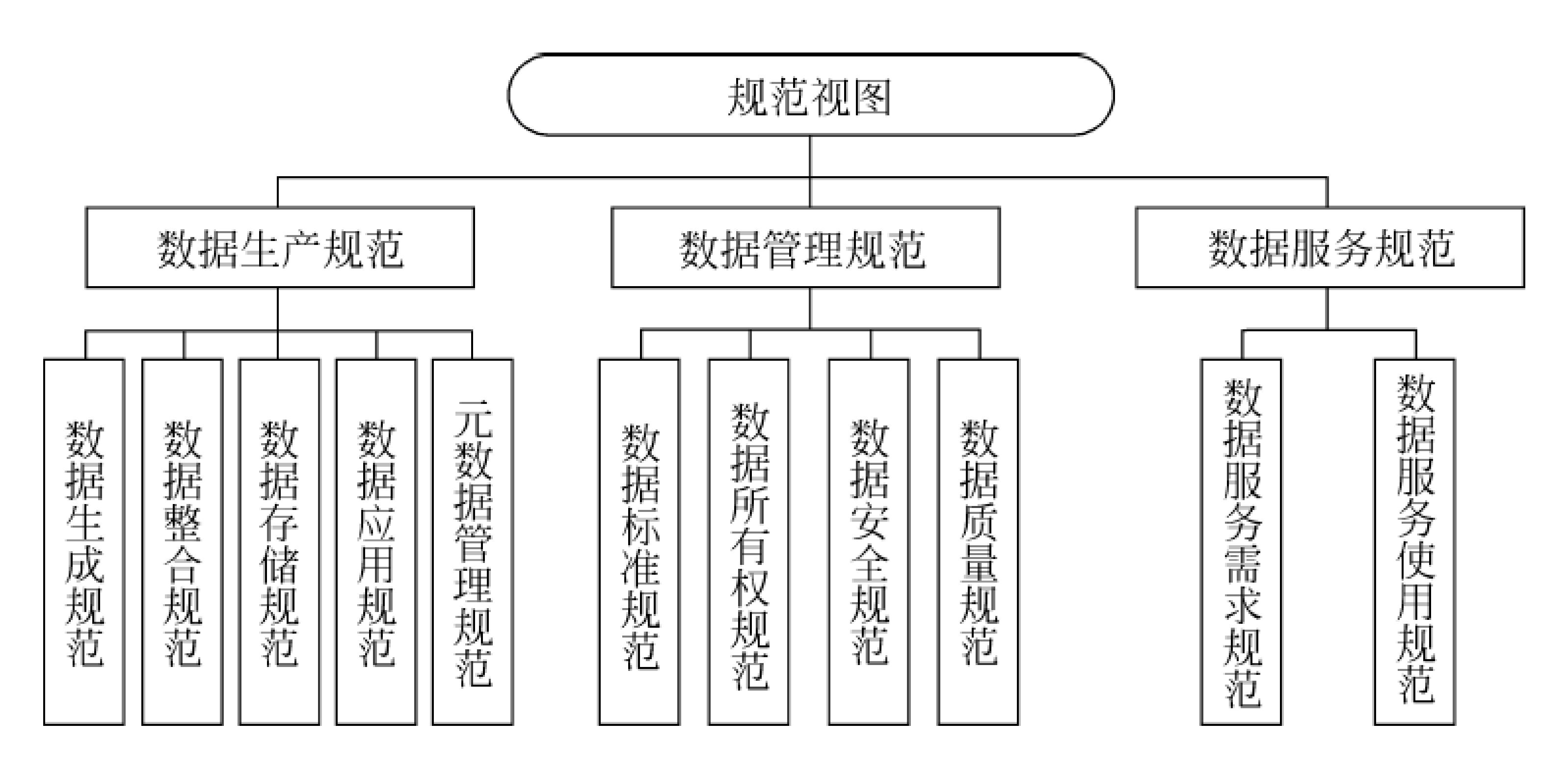


图 1.19　数据服务管理模型的规范视图

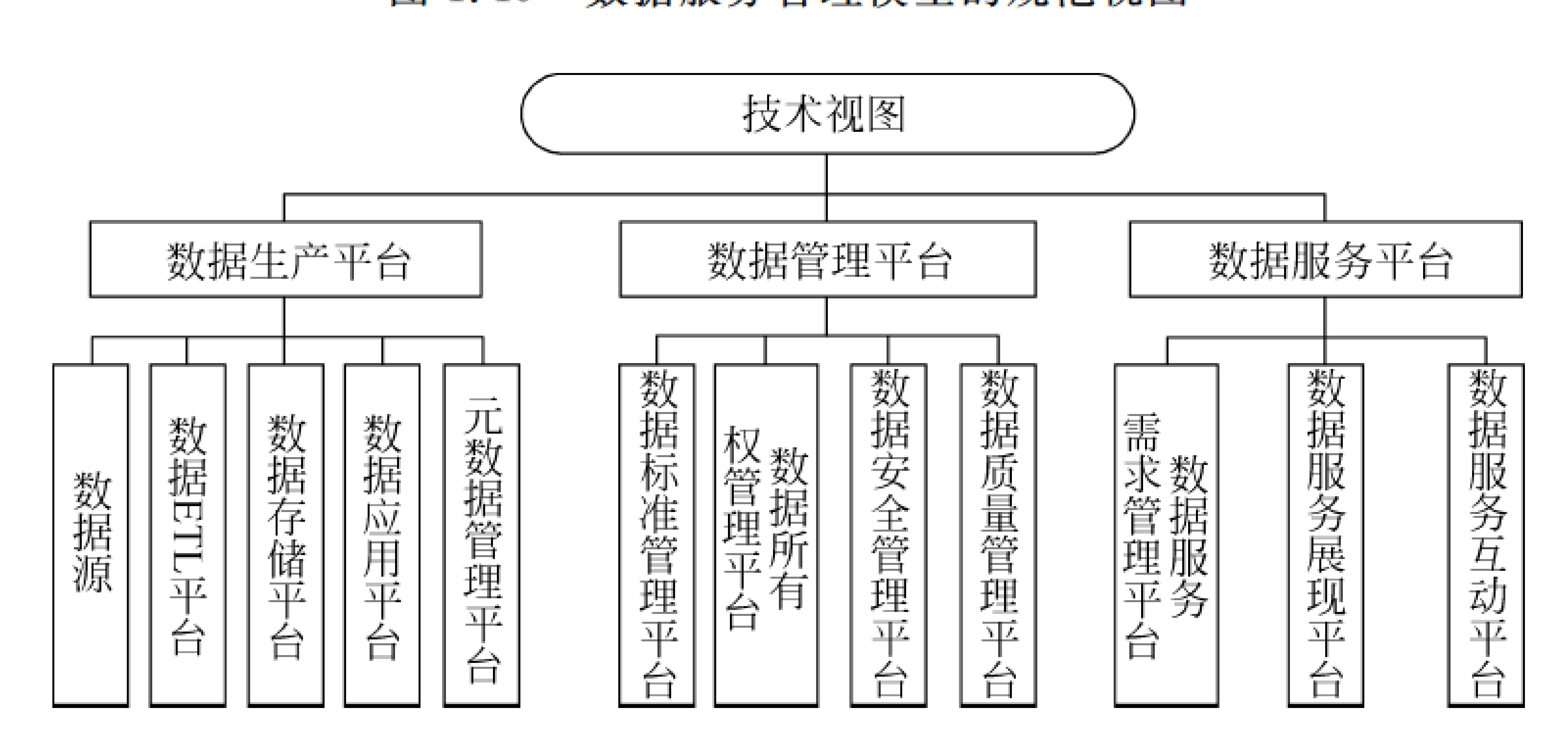


图 1.20　数据服务管理模型的技术视图

1.8.5　数据规划

数据管理薄弱，不仅影响工作效率，更造成大量资源的浪费，对企业的成本控制和快速发展都将造成十分不利的影响。主要体现在产生的数据无法准确地反映企业经营状况；多个业务部门对某个经营数据负责，其数据一致性、准确性无法保障；各业务部门对经营数据的理解不一致，在彼此沟通时产生歧义；无法对管理层的决策过程提供准确的数据支持。

数据规划是对企业的业务、管理活动中发生的数据进行分析、整理、提炼以描述出数据的构成，数据间层次结构、数据间相互关系，建立体系化数据资源体系的过程；是从企业数据流的视角对数据结构进行一个全面、系统的看待。主要解决“有哪些数据”、“数据之间是什么关系”的问题。规划后的数据体系应该具有以下特点：

- 先进性——规划后的数据体系应该符合当前的技术标准，适应企业 3～5 年的发展需要，即在 3～5 年之内具有先进性。
- 可扩展性——数据体系必须具有可扩展性，根据企业的需要对模型进行扩展，支持可持续发展。
- 可靠性——设计的数据体系必须准确可靠，能够保证基于这些数据体系的信息系统安全可靠地运行。
- 一致性——设计的数据体系在整个企业范围内是完全一致的，不能存在二义性，即在企业内部真正达到单一版本的事实(A Single Version of Truth)。

1. 方法

关于数据规划的方法,经历了不同的发展历程。最初是面向单一应用系统的数据规划,后来提出了基于信息工程理论的总体数据规划,通过业务调研与分析,建立业务模型,在此基础上分析数据流向,进行数据分析和数据建模,是一种模型导向的规划方法,旨在建立企业信息模型,作用是指导应用开发,协调数据交换。它需要的时间跨度非常长,短则三五个月,长则半年。后来为了满足企业业务管理、销售管理对数据的需要,提出面向业务需求的解决方案。面向业务需求的数据规划思路是,基于企业业务管理需求,从企业业务活动中产生的报表、视图入手,建立各级数据指标体系。对于一些数据对象,区分数据对象的静态属性和动态属性,例如一年的销售量、一个月的销售量、一个季度的利润等指标;对指标进行分解和分级,对最小数据单元进行归纳形成数据主题,如图 1.21 所示。

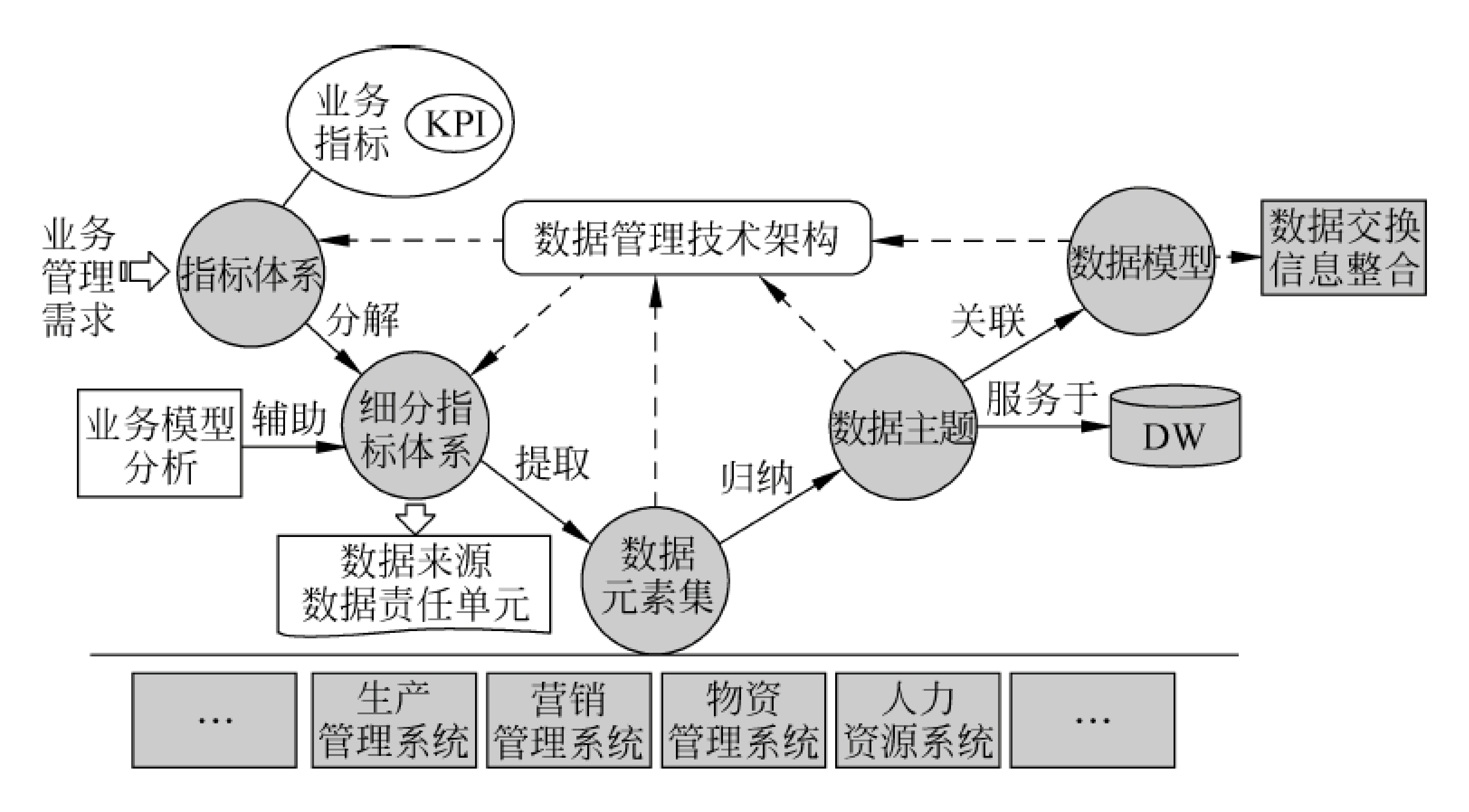


图 1.21　面向业务需求的数据规划示意图

这一思路以业务需求为导向,以业务过程中数据表单为主线分析构建数据资源体系。在指标体系的分析过程中,通过对业务体系的分析,进而加深对指标体系的正确理解和认识。然后,形成最小的数据单元并找出数据的来源和数据的责任来源,这样来源清楚、责任明确。在此基础上形成最小的单元数据主题,服务于数据管理。

2. 目标

通过数据规划,可以对以往结构不合理、冗余、混乱、分散或是未能收集整理利用的数据进行规范化的重组工作,从而尽可能地减少数据接口,实现基于高标准数据环境的系统集成。可以说,数据体系建设的根本任务,是在整个数据范围内建立统一的、稳定的和规范的数据模型。

在标准化的基础上,可以形成基于主题的高水平数据应用环境。在这样的数据环境中,数据是面向业务主题的,而不是面向单张报表的;信息是共享的,而不是信息私有或部门所有的;一项数据一次一处输入系统,而不是多次多处输入系统。数据规划的多重目标如图 1.22 所示。

3. 主要内容

数据规划的主要内容包括三个方面:一是对数据指标体系进行分析,了解业务指标的构成和结构;二是形成数据资源体系规划;三是形成数据管理技术架构规划。

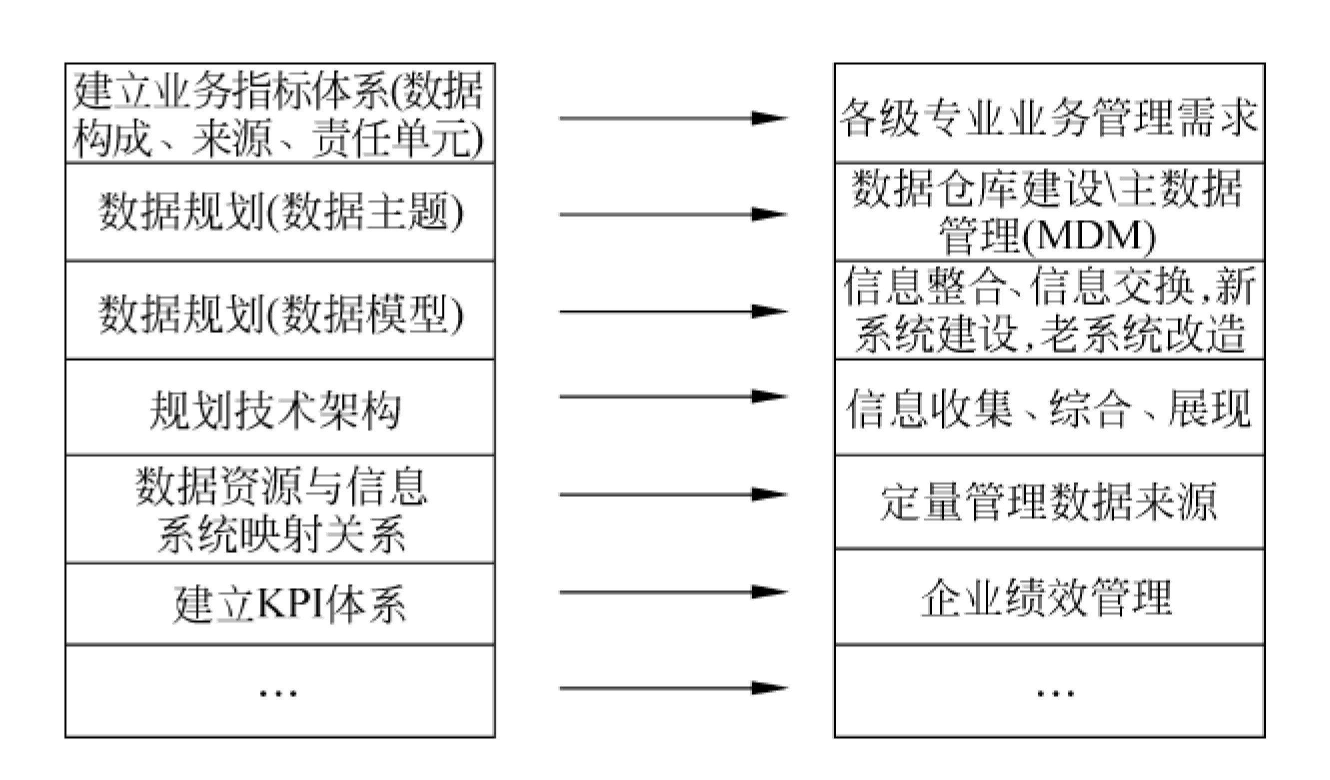


图 1.22　数据规划的多重目标

指标体系分析,包括业务与管理需求分析、业务体系分析、指标体系梳理和指标体系分解等。指标体系分析的思路首先是基于用户的视图和报表,这是基础,继而形成指标集,这时是没有分层次的。然后对指标分级分解,形成结构化的指标集。如图 1.23 所示为一级指标分解后成为二级指标的示例图。

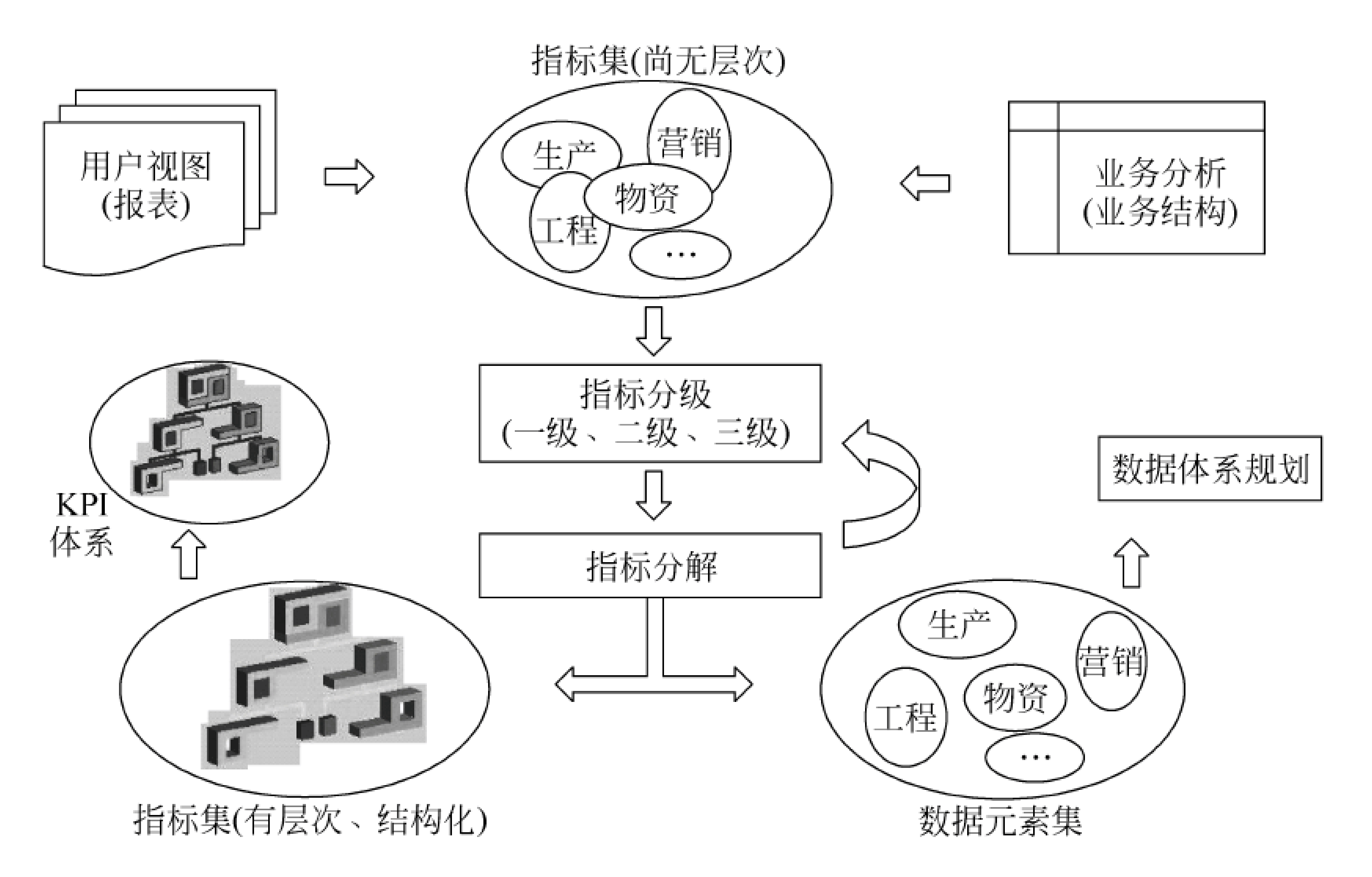


图 1.23　指标分解示意图

数据资源体系的规划是基于指标体系分析形成的最小的数据元素进行归纳形成数据主题,然后对数据主题之间进行关联分析,形成一个整体模型,它既包括一些静态的属性,也包含一些动态的指标。

当各类数据指标梳理出来后,则需要一个技术架构支撑。这部分工作包括数据资源能力评估,在此评估基础上提出数据需求分析,然后制订总体架构、功能架构和分层次技术架构。在技术架构规划中,还需要建立数据与信息系统的映射关系和数据关联分析,如图 1.24 所示。

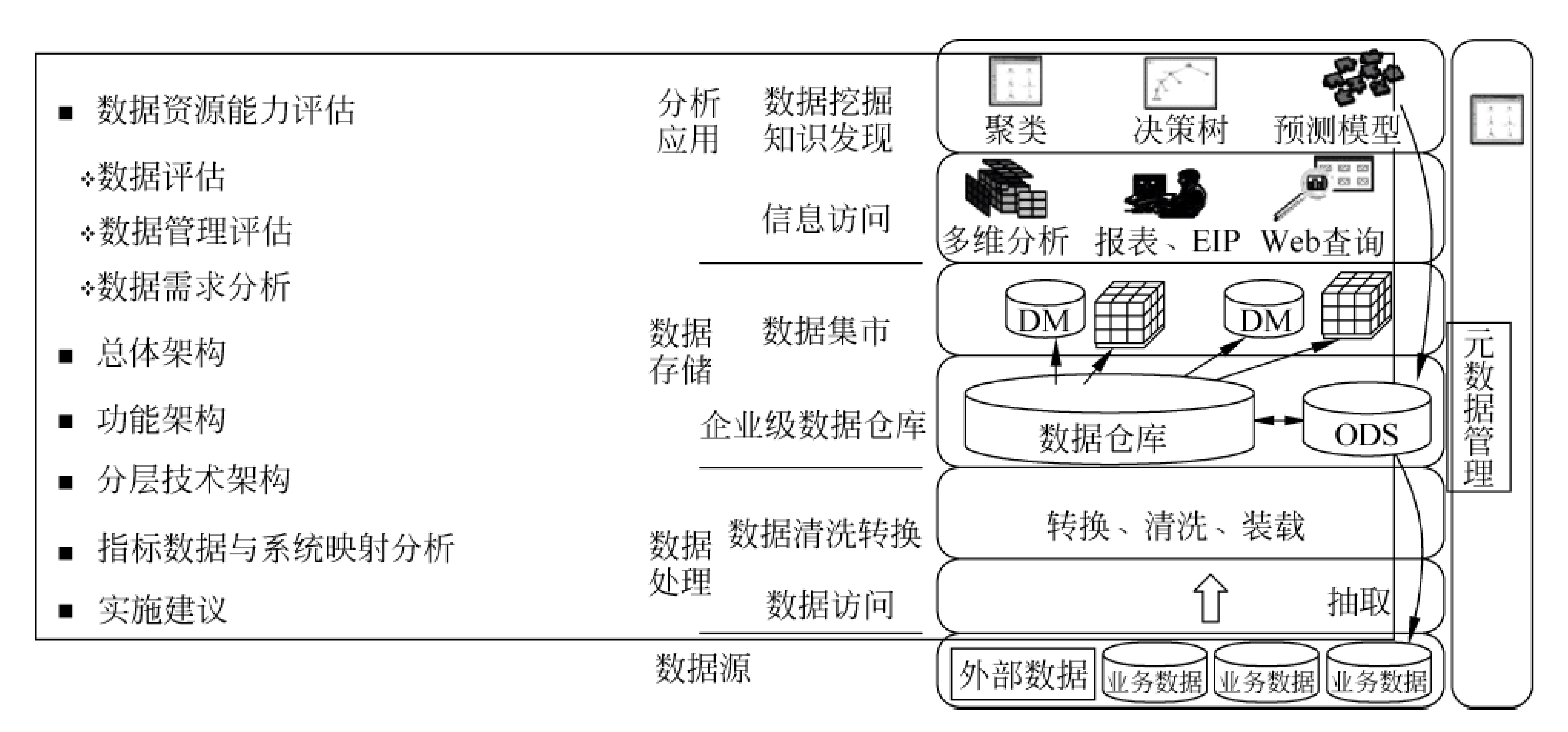


图 1.24 数据管理技术架构规划

1.8.6 技术方案

分析梳理这些影响数据质量的各类问题后，发现影响数据仓库的因素多种多样，要很好地控制数据质量，需要有效地管理这些影响因素，建立一套基于元数据管理的数据质量管理系统。本小节从元数据分类和管理、监控管理两方面阐述数据质量管理的技术方案。

1. 元数据分类和管理

1）元数据分类

元数据分为业务元数据、技术元数据和管理元数据三大类，每一类元数据由若干部分组成。

（1）业务元数据

业务元数据可划分为业务术语、业务描述、业务指标和业务规则四个部分，其中业务术语、业务描述和业务规则共同完成对业务相关信息的事实表述。数据质量管理通过对这些业务元数据的映射，可以得到对应的数据质量检查规则。每个项目可以根据自身业务需求对业务元数据主题进行扩充，扩充的元数据主题必须遵循 CWM 元数据扩展规范。

- 业务术语是一个名词或者名词短语，是对一种共识的定义，例如一个有效的账户被定义为此账户有余额且其有效期未过。
- 业务描述提供对各公司各项主要业务的统一描述，这部分内容应与数据仓库模型的主题信息、相关业务的统计指标建立关联，描述的内容包括：

① 业务基本描述。

② 业务管理规定，包括业务组织、业务流程、资费标准。

③ 业务统计。

业务描述是一个完整的描述，通过动词来连接术语使其成为一个有效的声明，例如开通具体业务的客户必须至少拥有一个有效的账户。

- 业务指标面向业务分析人员，是对业务关键信息的解释。业务指标的元数据信息主要包括：指标标识、指标名称、描述信息、建立时间、所属部门、创建者、类别和数据来源等。

- 业务规则是一种描述信息，定义和约束了部分业务逻辑，可以用于对业务逻辑结构进行验证，用于控制或者影响业务逻辑的行为。

(2) 技术元数据

以经营分析系统为例，技术元数据包含关于经营分析系统数据技术层面的信息，描述了数据源接口、ETL、数据仓库和数据集市、一级经营分析系统接口等子系统的数据特征。根据经营分析系统的特点及其数据质量管理的具体要求，技术元数据可分为数据源接口、ETL 过程、数据仓库和数据集市存储、分析应用、运行环境信息、运行状态信息和质量管理过程七个部分。下面对各部分所包含的实体加以说明。

① 数据源接口元数据

经营分析系统的数据来源主要是外部的操作型应用系统，经营分析系统管理的数据源接口元数据主要是关于 BOSS、客服、网管、智能网、DSMP、彩铃等外部系统与经营分析系统间接口的相关信息，包括描述源系统数据接口的定义信息、接口数据实体结构(主题信息、实体定义、实体内部列定义、关键字、索引定义等)、接口数据实体间关系。

② ETL 过程元数据

ETL 过程元数据管理内容主要是 ETL 规则，包括源系统数据到数据仓库的映射关系、ETL 程序结构信息、数据转换和清理规则等，主要包括：

- 接口抽取——描述数据抽取过程的相关信息，针对具体接口主要包括：

  接口抽取条件——接口抽取的前置条件和数据条件。

  接口抽取周期——日、月等。

  接口抽取时间——周期内的抽取时间。

  接口抽取方法——增量、全量等。

- 数据清洗、转换、装载——描述接口数据到仓库数据的处理过程信息，包括源系统数据到数据仓库的映射关系、ETL 程序结构信息、数据转换和清理规则等。

③ 数据仓库和数据集市存储元数据

数据仓库存储涉及 ODS、仓库底层数据和集市数据等基本信息，以及数据流处理逻辑相关信息的提取，具体包括：

- 数据仓库逻辑模型

逻辑模型是企业元数据的重要组成部分，其实现方式主要是将建模工具(如 ERWIN)中的模型信息纳入元数据管理中。

- 数据存储结构信息

数据存储结构反映数据仓库的物理实体信息。数据仓库内部的结构数据管理内容繁杂，主要包括 ODS 和数据仓库的数据结构、数据定义、物理数据模型结构、程序代码描述、数据库目录以及文件规划等。其中，数据库目录包括需纳入管理的表、关系以及索引和视图的定义等。

- 数据流及其处理逻辑

这部分内容描述整个数据仓库中数据的流向、数据的处理逻辑、数据仓库各应用程序模块接口等系统全局性信息。数据流向和数据处理逻辑信息是进行系统开发、维护、升级前和数据变更前的影响分析的重要信息源。

• 数据生命周期信息的存储和检索

数据生命周期主要记录数据在不同生命周期所存放的位置。通过元数据管理系统对不同类型以及不同生命周期的数据进行有效管理,支持对处于生命周期不同阶段的数据进行信息检索。

④ 分析应用元数据

在数据仓库或数据集市的基础上,经营分析系统提供了 KPI、OLAP、综合报表、数据挖掘、一级经营分析系统接口等应用,分析应用元数据主要包括:

- 指标技术定义信息——包含指标定义、对应维度定义、指标统计口径和指标关系等信息。
- 多维数据信息——主要指对多维数据相关信息的提取,主要包括:
  - 多维主题基本信息——包含主题与指标、维度等的关系;
  - 多维数据库信息——鉴于主流的多维数据库都支持 CWM 标准,针对系统的建设情况,可以考虑选用元数据工具进行相关信息的提取。
- 一级经营分析系统接口——作为数据分发过程与 ETL 处理过程对应,主要包括一级经营分析接口的定义、标准代码定义、数据转换规则以及其他涉及一级经营分析系统接口数据生成上传的元数据信息。
- 数据挖掘信息。
- 应用使用状况信息——主要记录各业务应用的使用频次、使用人数等信息。

⑤ 运行环境信息元数据

描述经营分析系统运行环境的相关信息,主要包括:

- 任务调度信息——经营分析系统处理涉及多个环节的大量任务,通常由一个统一的任务调度系统集中控制这些任务的执行,获取这些任务的相关信息对于了解和优化系统至关重要。主要提取信息包括任务基本信息、任务依赖信息和任务执行信息等。
- 系统运行环境信息——包括经营分析系统相关的主机信息、操作系统信息、文件系统信息、系统处理目录信息、数据库信息和表空间信息等。

⑥ 运行状态信息元数据

运行状态信息库是在数据质量检查点上产生的,描述系统运行情况的各种度量数据。主要包括数据量、数据处理周期、数据处理过程运行情况等。系统运行状态信息包括以下几类:

- 源数据状况——主要包括文件传送完整状况、文件记录合法状况、文件传送及时状况、文件加载正确状况等。
- ETL 运行状况——主要包括作业运行成功状况、作业失败信息、抽取和变换数据量、加载数据量及正确状况、作业调度和操作状况等。
- 数据仓库运行状况——主要包括实体主键取值合法状况、实体属性的完整状况、外键取值和引用合法状况、属性合法状况、数据汇总状况等。
- 数据集市运行状况——主要包括实体主键取值合法状况、实体属性的完整状况、外键取值和引用合法状况、属性合法状况、实体合法状况(属于某地市或部门)、数据汇总状况等。
- 前端运行状况——主要包括 OLAP 加载状况、OLAP 运行状况、OLAP 存储情况、应用运行合法和成功状况、应用满意状况等。

⑦ 质量管理过程元数据

质量管理过程元数据主要是根据数据质量管理的要求，为支撑数据质量检查的相关功能，从监控经营分析数据及系统运行情况角度，抽取的数据质量检查方法和标准。

质量管理过程元数据的规则有两个来源：一个来源是系统各个处理环节的技术元数据，反映的是系统技术层面的处理情况，如数据量和指标值是否合理，处理过程应该在什么周期内什么时间段内完成是正常，主机资源、表空间、文件系统的使用应该控制在什么样的阈值下系统处理不会有问题；另一个来源是业务规则、指标口径等业务元数据，通常会把业务规则的信息转化成技术化的规则、算法和度量标准，如指标间的稽核关系，而类似指标合理性的度量标准，则会根据业务发展情况而不断变化。主要包括：

- 规则库——规则是数据质量检查的具体标准，主要包括数据的约束规则和合理性规则，以及检查数据处理过程及处理环境是否正常的规则等。规则分为三类，即约束规则、计算规则和条件规则。

第一，约束规则描述了一种条件必须为真或假的强制规则。这种约束可以是结构化(Structural)约束，也可以是行为(Behavioral)约束。其中：

结构化约束——当创建术语或者改变术语之间的关系时，结构化约束能够保证术语的完整性。

行为约束——典型地被定义为“前置条件”和“后置条件”。只有在符合“前置条件”的情况下，操作才能够正确地执行；“后置条件”保证了操作结果的正确性，“后置条件”表示该行为是否满足了其预期的结果。例如，客户在开户时余额为0，在能够使用移动业务之前，必须先充值，这里前置条件就是必须先充值，而后置条件是指所办理的业务是否成功办理。

第二，计算规则描述计算关系，例如，月末余额＝上月末余额＋本月充值金额－本月消费金额。

第三，条件规则描述当条件成立时，触发相关的事件或事务，例如，预付费客户的账户余额小于0，则对其实施停机操作。

- 算法库——主要是指支撑上述检查规则所需要的基本算法的集合，结合具体的规则由数据质量检查相关功能调用。
- 度量信息——度量信息是数据质量检查的基本依据，它反映了数据质量的衡量标准，也是数据质量管理系统技术元数据的一个重要组成部分。

度量信息是经验的积累，需要在数据质量不断建设的过程中逐步精确化，随着衡量标准的不断细化和精确，经营分析系统数据质量也会不断提高。

(3) 管理元数据

管理元数据主要是指涉及开发、运维管理各方面的基本信息，强烈建议对管理元数据进行存储，在此基础上对系统需求开发和日常运维管理流程提供IT化支撑，从管理流程角度对数据质量提供保证。管理元数据主要包括：

- 管理流程定义——指需求开发、运维管理流程及其具体步骤、输入输出和参与角色的描述；
- 角色及职责定义——指参与需求开发、运维管理流程的角色及其具体承担职责的描述；
- 人员组织信息及工作内容分配——指需求开发和系统运维过程中各参与人员承担

的角色及具体负责的工作内容描述,例如具体某个需求开发由谁牵头负责,ETL 某个加载转换过程由谁负责维护等系统访问情况信息。

2) 元数据管理

数据质量管理系统的元数据作为基本支撑,方便用户对系统的理解,并在数据质量出现问题时,便于问题的定位。

(1) 元数据维护

元数据维护提供对元数据的增加、删除和修改等基本操作。另外,还应支持实体、实体间关系的建模,数据处理过程的描述等功能。对于元数据的增量维护,要求能保留历史的版本信息。

元数据的维护操作是原子操作,这些原子操作的实现需要借助于元数据管理平台提供的 JMI 接口或者 CORBA IDL 接口完成。

在实施元数据维护的过程中,需要对这部分操作的过程进行一定约束和限制。元数据定义为如下四种状态。

① 审核状态

对于需要进行维护的元数据,首先进入审核状态,从而等待审核人员的核查,确定该元数据是否需要进行修改。

② 公示状态

审核通过后,进入公示状态,即处于对元数据定义征求意见的状态,没有最后定稿,公示状态有一定的期限要求。

③ 发布状态

公示状态期限的限制到期后,根据在公示状态期间回馈的意见,给出元数据维护的准确方案,进入发布状态。

④ 维护状态

用户提交修改元数据请求后,相关元数据进入维护状态,对相关元数据的维护需要保留其历史版本信息。

(2) 元数据导入/导出

元数据导入/导出实现元数据的自动加载、批量导入/导出和模板化导入/导出,是元数据维护手工操作的重要补充。

① 自动加载

自动加载能够提供自动方式快速地从各种软件或存储(如数据库、OLAP 软件、前端展现工具)中提取元数据,并存储到元数据库中。这种方式通常采用 API 或 Web 服务直接连接各软件的元数据库。

② 批量导入/导出

批量导入/导出功能将按 XMI 文档格式组织的元数据批量导入元数据库,或者将元数据库中的指定部分导出为 XMI 文档。系统对导入/导出过程进行日志记录。元数据的 XMI 文档中描述元数据内容、属性,以及对新增、修改、删除动作的标记。用户通过执行批处理命令或屏幕菜单选择批量导入/导出元数据,并支持即时调用或后台定时自动执行。

③ 模板化导入/导出

模板化导入/导出是为非结构化、无法自动抽取或非 XMI 文档的元数据提供的批量导

入/导出支撑功能。通常采用预定义模板格式的元数据描述文档，整理出元数据的基本信息文档，通过该功能导入系统，并自动转化为元数据存储；同样也支持把元数据信息导出到预定义格式的文档中，并提供下载，方便交流。

对于不满足 CWM 标准的元数据(如大部分指标数据等)，通常需要借助 XMI 来定义相应的模板。该模板满足 CWM 标准，可以利用编写相应的程序实现这部分元数据的导入和导出。

(3) 同步检查

同步检查主要是指定期从系统中抽取元数据，并与元数据库的对应信息进行比较，及时发现系统的应用变更，保证元数据的及时同步更新。

从系统中抽取元数据主要分为两类：自动抽取和人工抽取。因此，有如下两类同步检查方法：

① 自动同步检查

对于需要检查的元数据，利用 API 或 CORBA IDL 接口对系统中的元数据进行直接查询和访问，获得相应的元数据，然后进行比较，从而确定相应的元数据是否保持同步。

② 人工同步检查

对无法进行自动抽取检查的元数据需要人工在源系统与元数据库之间进行比较，确定是否保持一致。

在同步检查过程中，值得注意的是：

① 同步检查时机

同步检查时机分为实时同步检查和定期同步检查两类。实时同步检查适用于可以自动抽取的元数据，触发检查的时机是源系统中的元数据发生了变更或是相应系统中的元数据发生了改变；定期同步检查适用于两种情况，即当用户手工修改源数据系统或者是元数据库时，则需要对元数据进行同步检查；或者对于整个数据质量管理系统而言，按照某个特定的周期(1 天或者 1 周等)进行元数据同步检查。

② 同步检查度量

元数据——同步检查的度量包括及时率、完整率和正确率三个方面。

- 及时率——指定类型的元数据在某一抽取检查周期内按一定时间要求抽取的元数据项和应抽取项的比率。
- 完整率——指定类型的元数据在某一抽取检查周期内抽取的元数据项和应抽取项的比率。
- 正确率——指定类型的元数据在某一抽取检查周期内抽取比对无误的元数据项和应抽取项的比率。

③ 同步检查过程

在同步检查发现差异时，原则上不能直接修改元数据，而是给出各类元数据的差异报告，并由数据质量管理员确认后，利用元数据维护工具进行元数据的更新。

(4) 实体查询

实体查询是指对元数据库中的数据实体基本信息的查询功能，通过该功能可以查询数据库表、维表、指标及其他纳入管理实体的基本信息，查询的信息按处理的层次和业务主题进行组织，实体查询返回实体及其所属的相关信息。实体查询要求能支持对历史版本信息

的查询,以了解具体实体的历史变更情况。

① 过程查询

过程查询是针对具体的数据转换、汇总等处理过程,查询其具体用途、参与的输入、输出实体等详细信息,从而帮助理解数据处理过程。过程查询要求能支持对历史版本信息的查询,以了解处理过程的历史变更情况。

CWM 标准中,有专门的仓库过程(Warehouse Process)和转换过程(Transformation Process)实体,过程查询可以直接返回过程本身的详细信息。

过程查询还可以基于过程实体本身包含的关联信息,得到与其关联的实体信息。

② 影响分析

影响分析(Impact Analysis,也称冲突分析)是指从某一实体出发,寻找依赖该实体的处理过程实体或其他实体。如果需要可以采用递归方式寻找所有的依赖过程实体或其他实体。该功能支持当某些实体发生变化或者需要修改时,进行影响实体范围的评估。

影响分析应能够以图形的方式展现所有实体和关联关系。

③ 血统分析

血统分析(Lineage Analysis,也称血缘分析)是指从某一实体出发,往回追溯其处理过程,直到数据输入的源头,即源系统接口数据。对于不同类型的实体,其涉及的转换过程可能有不同类型,例如:对于底层仓库实体,涉及的可能是 ETL 处理过程;而对于仓库汇总表,可能既涉及 ETL 处理过程,又涉及仓库汇总处理过程;而对于指标,则除了上面的处理过程,还涉及指标生成处理过程。除了数据接口实体由源系统提供,作为系统的数据输入,其他的数据实体都经过了一个或多个不同类型的处理过程。血统分析正是提供了这样一种功能,可以让使用者根据需要了解不同的处理过程,每个处理过程具体做什么,需要什么样的输入,又产生了什么样的输出。

为了实现血统分析,对于任何指定的实体,首先获得该实体的所有前驱实体,然后对这些前驱实体递归地获得各自的前驱实体,结束条件是所有实体到达数据源接口或者是实体没有相应的前驱实体。血统分析应能够以图形的方式展现所有实体和处理过程。

④ 实体关联度分析

实体关联度分析是从某一实体关联的其他实体和其参与的处理过程两个角度来查看具体数据的使用情况,从而形成一张关联所参与数据处理过程的网络,进一步可以了解该实体的重要程度。该功能可以用来支撑需求变更影响评估的应用。

对于关系数据表,可以分析与某个表有关的实体或处理过程,查看具体数据的使用情况,从而判断该表的重要程度。该功能有助于寻找数据库中关键表和冗余表,为系统调优提供支持。

在 CWM 模型的每个实体中,可以有一个单独的对象(或者类)函数,它可以在实体关联信息的支持下,返回该实体所关联的所有实体和参与的所有过程。

⑤ 实体差异分析

实体差异分析是对不同实体的元数据进行检查,用图形和表格的形式展现它们之间的差异,包括名字、属性及数据血统、对系统其他部分影响的差异等。

系统中存在许多类似的实体。这些实体(如数据表)可能只在名字上或者是在属性中存在微小的差异,甚至有部分属性名字都相同,但处于不同的应用中。由于各种原因,这些微

小的差异直接影响了数据统计结果，需要清楚了解这些差异。该功能有助于进一步统一统计口径，评估近似实体的差异。

(5) 版本管理

版本管理是对不同时期进入元数据库的同一实体的元数据进行管理。基本功能是能够显示同一实体的元数据的修改历史。另外还提供版本差异分析、版本变更分析等。

变更通知是当元数据发生改变时，系统自动发信息(邮件、短信)给订阅用户。用户可以主动订阅自己关心的元数据，帮助了解与自身工作相关的业务系统变更情况，提高工作的主动性。

2. 监控管理

在数据质量管理系统中，数据质量监控是基础功能。数据质量监控实时采集数据处理过程的各个质量监控点上的质量信息，执行数据质量监控流程，进行数据稽核诊断，然后直观展现检查结果。数据质量管理人员可根据告警信息，采取相应的质量控制措施。最后，还应评估总结本次数据质量监控活动的有效性和及时性，以利于以后数据质量管理工作的改进。

在数据质量监控中对质量问题的监控、诊断和处理结果应整理形成质量问题处理案例，并记录到知识库中，以便进行知识共享。

数据质量监控流程是数据质量监控模块的重要组成部分，其监控流程如图1.25所示。

数据质量监控流程包括采集、检查、报告、处理和总结五个主要环节。

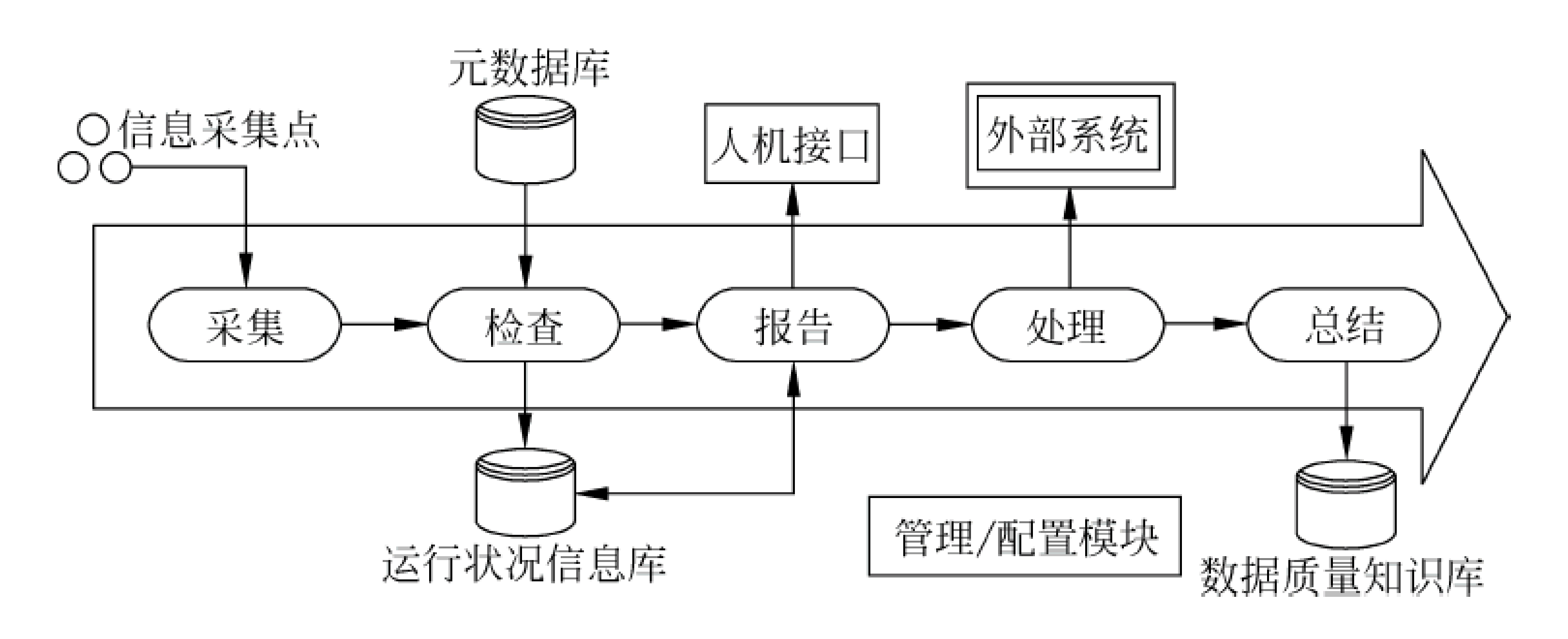


图1.25 数据质量监控流程

1) 采集

数据质量信息采集需要注意两方面的问题，一方面尽量将信息采集点前移，将信息采集点前移到省级经营分析系统与BOSS等外部业务系统的接口上，及时发现数据传输中的问题，不但可以保障省级经营分析系统的数据质量，同时也可进一步保障一级经营分析系统的数据质量；另一方面信息采集点覆盖数据处理的主要过程，经营分析系统的数据处理包含多个过程，要求在主要过程中部署信息采集点，支持对数据质量的全局监控。

经营分析系统数据处理包含多个过程，如外部业务系统接口传输、ETL、数据仓库处理、数据集市处理、一级经营分析系统接口传输和前端应用展示等。各个主要环节均应根据实际情况设置信息采集点，如图1.26所示。

根据已有系统的实际情况，以及具体的监控需求和配置，信息采集点可以采集以下类型的信息用于数据质量检查：

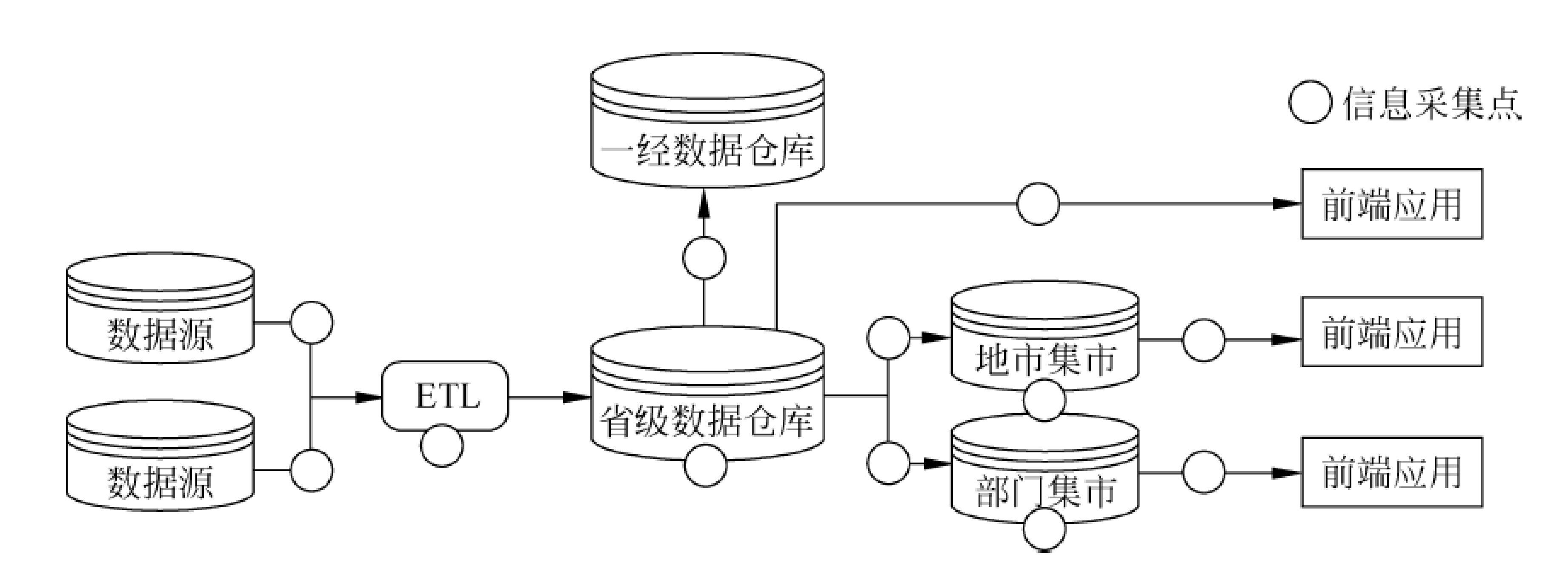


图 1.26　经营分析系统信息采集点分布图

- 数据处理日志
- 部分数据内容
- 校验文件
- 其他

2）检查

数据稽核是数据质量检查的基本功能，是根据元数据库中预先配置的规则、算法和质量检查度量，从数据的正确性和合理性等多角度进行检查，以及时发现数据质量问题。以经营分析系统为例，数据稽核包含：

（1）接口文件级检查

接口数据是经营分析系统数据的源头，接口数据质量问题的改进是经营分析系统数据达到较高的数据质量的最基本保障。对于每个到达的源数据文件，数据质量检查环节首先进行的是文件级检查。文件级检查主要包括：

- 文件传送及时率
- 文件传送完整率
- 文件记录合法率
- 文件加载正确率

此外，文件级检查还需做以下方面的检查：

- 校验文件名称是否正确
- 校验文件是否可以正常打开
- 接口数据文件名称是否正确
- 接口数据文件是否存在
- 接口数据文件是否可以正常打开
- 接口数据文件记录总长度是否与接口规范中的总长度一致
- 接口数据文件大小是否与校验文件中大小一致
- 接口数据文件数据日期是否与校验文件中数据日期一致

（2）仓库数据检查

仓库数据检查从仓库模型约束、实体属性、实体关系和实体业务特征等角度，根据预定义规则，检查数据合理性，及早发现异常，保证仓库对分析、报表等所提供数据的可用性与正确性。主要包括：

- 实体主键取值合法性——对于数据仓库中的表而言，主键是记录的唯一标识，主键上不允许出现重复数据。主键检查主要是检查主键字段的值是否唯一。
- 外键检查——检查是否满足引用完整性的规则。
- 值域检查——检查特定字段的取值是否落在预定的取值范围之内。
- 编码规范检查——数据仓库中，对于一些字段值往往有一些特定的编码，编码规范检查按照检查规则配置中的编码规则进行检查。
- 实体关键属性的完整率——实体关键属性需要填充信息，不可为空。
- 属性合法性——实体属性的取值在语法和语义上均应符合业务逻辑。

（3）关键指标检查

经营分析系统需要进行检查的指标主要分布在经营分析系统的 KPI、OLAP、报表和一级经营分析系统接口等应用中。目前系统主要关注的指标类型如下：

- 客户分析类指标
- 业务量类指标
- 收入类指标
- 大客户类指标
- 合作服务类指标
- 市场竞争类指标
- 服务类指标

主要的检查方法分为：

- 数值检查——主要通过对单个指标值的监控来发现数据的异常、突变等情况。通过对指标与阈值上下限的比较进行检查。适用变化趋势平稳的业务关键指标，对于波动变化较大的业务数据，不建议采用此种方法检查。
- 波动检查——通过对单个指标值一段时间内的数值变化情况来检查数据的波动、变化情况。对于系统内的指标，分为月指标和日指标。其中，月指标支持与上月比较，日指标可以与前日、上周和上月进行波动比较。通过对指标不同周期之间的增减幅度与阈值范围上下限的比较进行检查。阈值范围上下限是一个数值区间，一般在－1～1 之间取值。适用于主要关注的业务发展指标。
- 扩展检查——通过对两个指标（也可以是一个指标的两个周期）按某几个维度展开后的增减幅度来发现数据的波动、变化情况。该检查的两个指标分别命名为左指标和右指标，即第一个选择的为左指标，后选的为右指标。通过对左右指标不同维度切面的增减幅度与阈值范围上下限的比较进行检查。阈值范围上下限是一个数值区间，一般在－1～1 之间取值。适用于需要进行维度细分检查的指标，如检查某项业务在省内各地市发展的一致性，相关用户数和收入是否同增同减等。
- 比较检查——通过对若干个指标值的简单四则运算（加、减、乘、除）来检验各个指标间潜在的平衡或其他比较关系。在这一简单四则运算中，可以带入常量运算，支持括号，适用于需要进行相关性检查的指标。

（4）处理过程检查

处理过程检查是指根据预先配置的规则和质量检查度量，对数据处理过程的稽核检查。重点关注处理过程的及时性，即根据预先设定的时间范围，从经营分析系统各数据处理环节

的处理效率角度进行检查,得出处理超出规定时间范围的接口及其具体的处理环节。处理环节包括接口数据的抽取、传输、清洗转换、加载、集市数据汇总生成和应用展现等。

数据采集点和质量检查点集成在一起,在数据采集的本地直接进行质量检查,分析其中存在的数据质量问题,如图1.27所示。

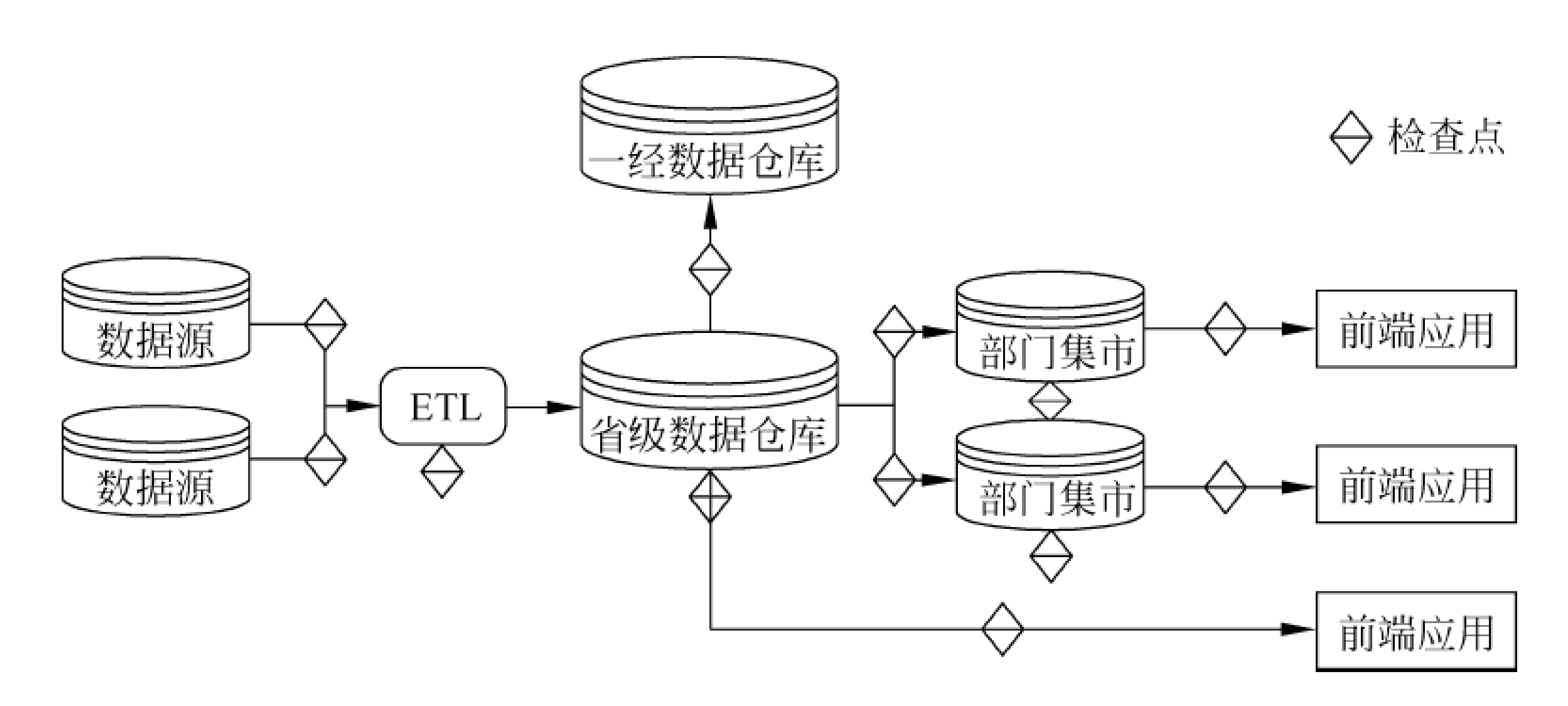


图1.27　数据质量检查点的部署图

数据质量检查点进行检查操作的依据是元数据库中预先配置的考核标准、规则、算法和质量检查度量等。检查点需要与元数据库交互,获取相关检查规则。检查点与元数据库的交互方式可以根据实际情况参考以下方案:

- 检查点主动读取元数据库——检查点根据自身的配置信息,主动访问并获取元数据库中与本检查点相关的规则数据。
- 利用管理配置功能远程配置检查点——数据质量监控模块具有管理配置功能,可以提供集成的系统管理配置界面,集中管理元数据库和分布式检查点,并将适当的检查规则分配到特定的检查点上,实现检查点的远程配置。

为了提高系统性能,检查点应对检查规则进行本地缓存,并根据配置信息,以定时或事件触发的方式更新本地缓存。

3) 报告

检查结果报告环节根据数据质量检查环节的结果信息,以规范的格式和特定的展现方式向数据质量管理人员报告数据质量情况。可以根据系统配置,分布式或集中式检查点从不同角度生成不同类型的检查或告警报告,并由检查点将检查报告存储于运行状况信息库中。

多数情况下,质量管理人员只关注存在质量问题的报告,这些报告通常以告警形式出现。

报告功能是数据质量管理的重要手段和方法,包括报告定义、报告生成与报告提示等功能。可以将检查结果报告从不同角度进行分类,例如:

(1) 按照问题严重等级分类可划分为提示(Information)、报警(Warning)、严重(Serious)和致命(Fatal)。

(2) 按照管理层次分类可划分为事件管理、问题管理和服务水平管理。其中,事件管理是在数据质量事件的粒度上反映和考查数据质量情况;由于多种数据质量事件常可归结为特定的数据质量问题,故问题管理是在问题的粒度上对数据质量情况的反映和考查;因为服务水平管理常涉及多种数据质量问题管理,所以服务水平管理是在服务的粒度上对数据质量情况的反映和考查。

告警是一类重要的检查报告，包括报警、严重、致命等级别，是数据质量管理人员最为关注的报告类型。监控系统的监控界面要求简洁醒目，能够形象直观地表示系统问题故障。维护人员可以根据任意字段对结果进行排序。

对于所有的告警信息，监控系统的告警形式可包括声音、短消息和电话。同时，告警形式可以具有不同的级别，如一级：声音，二级：短信，三级：电话。

各种告警形式的具体描述如下：

- 声音——可根据预先定义的告警级别通过声音进行告警。
- 短信——可根据预先定义的告警级别通过短消息进行告警。
- 电话——可根据预先定义的告警级别通过电话进行告警。

所有的告警信息都支持告警升级，如果告警信息超过告警升级的时间阈值尚未解决，告警信息自动升级告警级别，例如，报警升级到严重。

4）处理

在检查过程中发现数据质量问题时，系统需根据问题的严重级别启动相应的问题处理过程。问题处理过程可分为两大阶段。

第一阶段：数据处理流程挂起，问题隔离。当发生严重或致命级别的问题时，检查处理模块直接根据预先定义的处理方案，将数据处理流程挂起，并将存在问题的数据进行隔离。通过流程挂起和问题数据隔离，可将问题的影响范围控制在较小的范围内，防止问题放大，便于问题的解决。

第二阶段：问题分析，问题处理。当发生数据质量问题时，检查处理模块通过报告发布模块发出数据质量问题报告，处理分析模块对问题进行分析，生成问题解决方案，并通过适当的途径解决问题。

对于严重或致命级别的问题，需要执行以上两个阶段的问题处理过程；对于提示或报警级别的问题，则无须执行第一阶段的操作，仅执行第二阶段的操作即可。

5）总结

当问题处理环节结束后，数据质量监控需要启动总结环节，对问题处理的全过程进行记录和总结。总结可划分为不同的类型，即：

- 数据质量事件总结
- 数据质量问题总结
- 数据质量总体情况总结
- 数据质量阶段性总结
- 其他总结

第 2 章　数据仓库设计和实现

2.1　数据仓库设计

数据仓库是一个面向数据分析处理的数据环境，数据仓库的数据具有四个基本特征，即面向主题的、集成的、不可更新的和随时间变化的。这些特点说明数据仓库从数据组织到数据处理与传统的数据库存在很大区别，数据仓库系统设计与数据库系统设计的不同主要表现在以下几个方面：

（1）面向的处理类型不同。操作型数据库系统设计是建立一个操作型数据环境，其设计方法是面向应用的。即一般是从具体应用出发来进行数据库设计，然后在数据库上建立这些应用。数据仓库系统设计则是面向分析的，往往是从最基本的主题开始，不断地扩展新的主题，完善已有的主题，最终建立一个面向主题的分析型数据环境。

（2）面向的需求不同。面向应用的数据库系统设计具有比较明确的应用需求，这是数据库系统设计和开发的出发点和基础。在数据仓库环境下，不存在操作型环境中固定的且较确切的物流、数据处理流和信息流。数据分析处理的需求更灵活，没有固定的模式，甚至用户自己也对所要进行的分析处理不甚明了，因而在进行数据仓库系统设计时，很难获得对用户需求的确切了解，这就决定了不可能从用户需求出发进行数据仓库的设计。

（3）系统设计的目标不同。设计数据库系统时，事务处理的性能（主要表现为事务处理的响应时间）是系统设计的一个主要目标；而设计数据仓库系统时，更应该关心的是建立起一个全局一致的数据环境，作为企业决策支持系统的基础，因此数据仓库设计的一个主要目标是保证数据的四个基本特征，保证数据的全局一致性，实现对企业数据的全局管理和控制。

（4）数据来源或系统的输入不同。操作型环境的数据输入通常来自于组织外部，设计操作型数据库即是设计如何通过与外部交互获取数据，如何将获取的数据以适当的方式进行存储，如何对数据进行联机查询、更新等操作，以及如何保证数据的安全可靠与正确有效等。而数据仓库的数据主要来自于已有系统内部，设计数据仓库即是设计如何从现有的数据源中获取完整一致的数据，如何将所获取的数据进行转换、重组和综合，如何有效地提高数据分析的效率与准确性等。

综上所述，数据仓库的设计主要包括两个方面——与操作型系统接口的设计和数据仓库本身的设计。从某种程度上而言，“设计”并不能精确描述在启发方式下构建数据仓库时发生了什么。首先，载入一部分数据，供 DSS 分析员使用和查看；然后，根据最终用户的反馈，在数据仓库中修改、增添一些数据。这种反馈循环贯穿于整个数据仓库的开发过程。那种认为在构建数据仓库时，采用过去曾使用的设计方法就可以满足需求的想法是错误的。在数据仓库部分载入并且为 DSS 分析员使用之前，数据仓库的需求是不可能知道的。因此，设计数据仓库时不能采用与设计传统的“需求驱动”系统同样的方法。另一方面，那种认为不预测需求是好思路的想法也是错误的。实际上通常介于两者之间。

概括来说，数据仓库的设计主要包括：

- 体系结构设计
- 数据仓库模型设计
- 数据装载接口设计
- 数据仓库管理
- 元数据管理

(1) 体系结构设计

根据在业务探索和信息调研中所了解的用户业务环境和IT环境，设计数据仓库的整体架构，确定数据仓库的位置、网络需求、用户访问数据仓库的方式等。体系结构设计是对建立一个数据仓库系统的总体描述，从宏观和整体角度对数据仓库系统的各组成部分进行总体设计，并确定在设计过程中应遵循的总原则，保证数据仓库各个部分在开发过程中能够依据同样的基础和标准，在运行过程能够相互协调配合。后续的数据转换、应用开发、系统管理等工作将参照体系结构的设计和指导原则进行。

(2) 数据仓库模型设计

数据仓库模型设计包括概念模型设计、逻辑模型设计和物理模型设计三部分。首先进行的是概念模型设计，以确定数据仓库的主要主题及相互关系。概念模型设计主要完成：

- 界定系统边界，即进行任务和环境评估、需求收集和分析，了解用户迫切需要解决的问题及解决这些问题所需要的信息，需要对现有数据库中的数据有一个完整而清晰的认识。
- 确定主要的主题域，即确定系统所包含的主题域，然后对每个主题域的公共码键、主题域之间的联系、充分代表主题的属性进行较为明确的描述。数据仓库中的概念模型设计经常采用E-R模型和面向对象的分析方法。

逻辑模型设计是按照企业的业务规则和流程将各种数据有机地集成在一个完整的逻辑模型中。逻辑模型包括各个业务实体、业务实体的属性，以及业务实体之间的关系等。通过逻辑模型设计，可以对每个主题的逻辑实现进行定义，并将相关内容(如适当的粒度划分、合理的数据分割、增加的派生字段、记录系统定义等)记录在数据仓库的元数据中。

物理模型设计主要解决数据的存储结构、数据的索引策略、数据的存储策略、存储分配优化等问题。其主要目的一是提高性能；二是更好地管理存储数据。访问的频率、数据的容量和存储介质的配置都会影响物理设计的最终结果。

(3) 数据装载接口设计

数据装载接口即载入程序，可实现数据装载和数据综合功能。数据装载功能负责数据抽取、转换、清洗和集成；数据综合功能负责将集成的细节数据转化为不同综合层次的数据。

(4) 数据仓库管理

数据仓库管理负责安全和特权管理；跟踪数据更新；数据质量检查；管理和更新元数据；审计和报告数据仓库的使用和状态；删除数据；复制、分割和分发数据；备份和恢复；存储管理。

(5) 元数据管理

元数据为访问数据仓库提供了一个信息目录(Information Directory)，这个目录全面地描述了数据仓库中有什么数据、如何获取以及访问这些数据，是数据仓库运行和维护的中

心,数据仓库服务器利用它存储和更新数据,用户通过它了解和访问数据。元数据通常存储在专用的数据库中,该数据库可视为一个"黑盒",外部无法知道这些工具所用到和产生的元数据是如何存储的。此外,还有一类被称为元数据知识库(Metadata Repository)的工具,它们独立于其他工具,为元数据提供一个集中的存储空间,如 Microsoft 的 Repository、CA 的 Repository、Ardent 的 MetaStage 和 Sybase 的 WCC 等。

2.1.1 设计方法

在操作型环境中,业务过程和规则比较规范且固定。设计人员能够清晰地了解应用需求和数据流程,系统设计一般采用系统开发生命周期(System Development Life Cycle, SDLC)方法。而在分析型环境中,DSS 分析员一般是企业的中上层管理人员,他们对决策分析的需求不能预先做出规范说明,只能给设计人员一个抽象、模糊的描述。这就要求设计人员在与用户不断的交流过程中,将系统需求逐步明确和完善。人们为了突出这种需求不确定的开发过程,将数据仓库的设计方法描述成数据仓库环境下的系统开发生命周期(Cycle Life Development System,CLDS)方法,恰恰与 SDLC 相反,CLDS 是典型的数据驱动的,而 SDLC 是典型的需求驱动的,如图 2.1 所示。

数据库系统设计常用的 SDLC 方法有独立的收集需求和分析需求的阶段,而数据仓库系统设计采用的 CLDS 方法没有这样的独立阶段,而是将需求分析的过程贯穿在整个设计过程中。因为联机事务处理的需求是较为固定的,如商场的库存管理、采购业务等都有一定的操作规程,所以在一个系统开发生命周期内,系统需求在设计的前期阶段即收集需求和分析需求阶段结束后就应该确定下来,一旦进入 SDLC 方法的第三步构建数据库,如图 2.1(a)所示,系统需求就基本不变。而 CLDS 方法则要求在整个系统开发过程中完成对系统需求的收集、分析和理解。

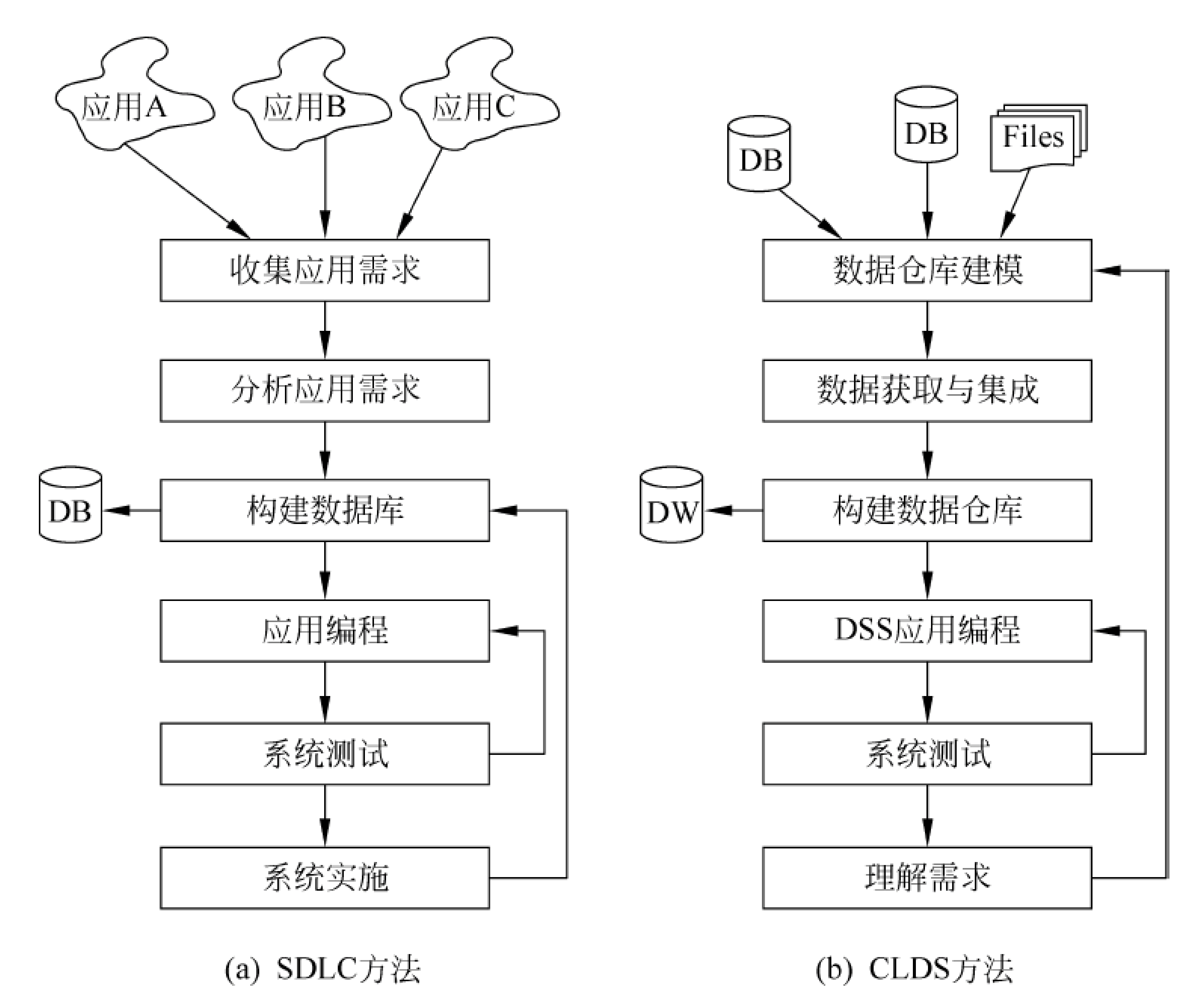


图 2.1 SDLC 方法与 CLDS 方法

数据仓库的设计方法是“数据驱动”的，其思路是利用以前所取得的工作成果进行系统建设。要充分利用现有的工作成果，唯一的办法就是能识别出当前系统设计与系统设计已完成工作的“共同性”。即在数据仓库系统设计前，需要清楚原有的数据库系统已经完成什么，以及它们对当前系统设计的影响等。应尽可能地利用现有的数据、代码等，而不是什么都从头开始。从源数据出发，分析数据，为新应用(分析处理)所用就是“数据驱动”的出发点。

“数据驱动”的系统设计不再面向应用，而是从已有的数据库系统出发，按照分析领域对数据及数据之间的联系重新组织数据仓库中的主题。

“数据驱动”设计方法的核心是利用数据模型有效地识别现有数据库中的数据和数据仓库中主题的“数据的共同性”。

2.1.2 体系结构设计

数据仓库开发是一个不断循环、反馈而使系统不断扩展、完善的过程，这对系统体系结构设计提出了很高的要求，要求体系结构具有良好的可扩展性和灵活性，能适应复杂多变的业务需求，不做或少做无效、重复工作。其次，数据仓库建设的目标不是数据集成，而是通过数据集成为业务发展提供前所未有的决策支持。因此，在数据仓库体系结构设计中应充分考虑到这一点，即结合业务应用的需求。

目前，比较成熟的数据仓库体系结构主要有两种：一是企业信息工厂(Corporate Information Factory，CIF)，创始人是数据仓库之父 Inmon；二是多维体系结构(Multidimensional Architecture，MD)又称总线架构(Bus Architecture)，创始人是数据仓库领域中颇具实践经验的 Kimball。

1. 企业信息工厂

企业信息工厂主要包括集成转换层(Integrated and Transformation Layer)、操作数据存储(Operational Data Store，ODS)、企业中心数据仓库(Enterprise Data Warehouse，EDW)、数据集市(Data Mart)和探索仓库(Exploration Warehouse)等部件，它们有机地结合在一起，为企业决策支持提供服务。

集成转换层是将来自操作型源系统的数据集成并转换到数据仓库，通常由一组程序组成，而其他部件如数据仓库和数据集市等则主要由数据组成。当业务数据来源多、业务复杂时，集成转换层建立一些临时表，为数据处理提供方便。此时，集成转换层包括程序和数据，也称数据准备区(Data Staging Area)。通常地，中等规模及以上的数据仓库系统都会建立数据准备区。

ODS 是建立在数据准备区和数据仓库之间的一个部件，用来满足企业集成的、综合的操作型处理需要。例如，提供尽可能实时的和集成的报表等。一般地，ODS 用以满足企业战略决策的需要，为可选部件。

企业中心数据仓库是 CIF 的核心部件，用来保存整个企业的数据。一般地，数据仓库用以满足企业战略决策的需要，其数据来自数据准备区和 ODS。

数据集市是为了满足企业特定部门的分析需求而专门建立的数据集合。数据集市的数据来源是数据仓库。CIF 的数据集市一般而言是非规范化的、定制的和汇总的。而多维体

系结构中的数据集市分为两种，即原子数据集市和聚集数据集市。一般而言，CIF 的数据集市相当于多维体系结构中的聚集数据集市。

探索仓库或数据挖掘仓库的建立主要是为了解决大型查询，提高数据仓库的效率。当有探索或挖掘需求时，会从数据仓库导出一部分数据供其操作。

CIF 的实现方式是，首先进行企业的数据整合，建立企业中心数据仓库即 EDW。对于各种分析需求再建立相应的数据集市或者探索仓库，其数据来源于 EDW。CIF 的数据流向一般是从源系统到数据准备区到操作数据存储到数据仓库到数据集市。当分析人员在数据仓库或数据集市中获得分析结论后，将有信息的回流。这种信息回流有可能是物理数据的回流，也可能是直接改变业务部门的决策。总之，要将分析的结果应用起来。通过这种信息回流，企业信息工厂的不同部件可以不断地相互调整，最终获得一种平衡。

2. 多维体系结构

多维体系结构主要包括后台(Back Room)和前台(Front Room)两部分。后台也称为数据准备区，是多维体系结构的核心部件。它是一致性维度的产生、保存和分发的场所。同时，代理键也在后台产生。前台是多维体系结构对外的接口，包括两种主要的数据集市：一种是原子数据集市；另一种是聚集数据集市。原子数据集市保存着最低粒度的细节数据，数据以星型结构进行存储；聚集数据集市的粒度通常比原子数据集市大，与原子数据集市一样，聚集数据集市也是以星型结构存储数据。前台还包括像查询管理、活动监控等为了提高数据仓库的性能和质量的服务。多维体系结构中，首先在数据准备区建立一致性维度，建立一致性事实的计算方法；其次在一致性维度、一致性事实的基础上逐步建立数据集市。每次增加数据集市，都会在数据准备区整合一致性维度，并将整合好的一致性维度同步更新到所有的数据集市。这样，所有建立的数据集市合在一起就是一个完整的数据仓库。

3. 比较

CIF 对于建立复杂应用，如挖掘仓库和探索仓库提供了更好的支持。但其建设周期比较长，成本较高。MD 中心数据仓库以多维模型保存，对于特殊的非维度型分析应用存在局限性。总之，这两种体系结构都是不错的选择，但各有优缺点。一种比较流行的做法是联合使用，即建立 CIF 的数据仓库和 MD 的数据集市。

4. 实例

目前，企业采用的典型数据仓库体系结构分为数据源、数据的存储与管理、OLAP 引擎、前端工具和应用四个层次，如图 2.2 所示。

(1) 数据源是数据仓库系统的基础，是整个系统的数据源泉。通常包括企业内部信息和外部信息。内部信息包括存放于 RDBMS 中的各种业务处理数据和各类文档数据；外部信息包括各类法律法规、市场信息和竞争对手信息等。数据仓库可通过 ODBC、JDBC 和 OLE-DB 等多种标准接口与这些数据源互连。

(2) 数据的存储与管理是整个数据仓库系统的核心和关键。数据仓库的组织管理方式决定其有别于传统的数据库，同时也决定了其对外部数据的表现形式。数据仓库针对现有各业务系统的数据，进行抽取、清理并有效集成，按照主题进行组织。数据仓库的组织形式按照数据覆盖范围可以分为企业级数据仓库和部门级数据仓库(通常称为数据集市)。

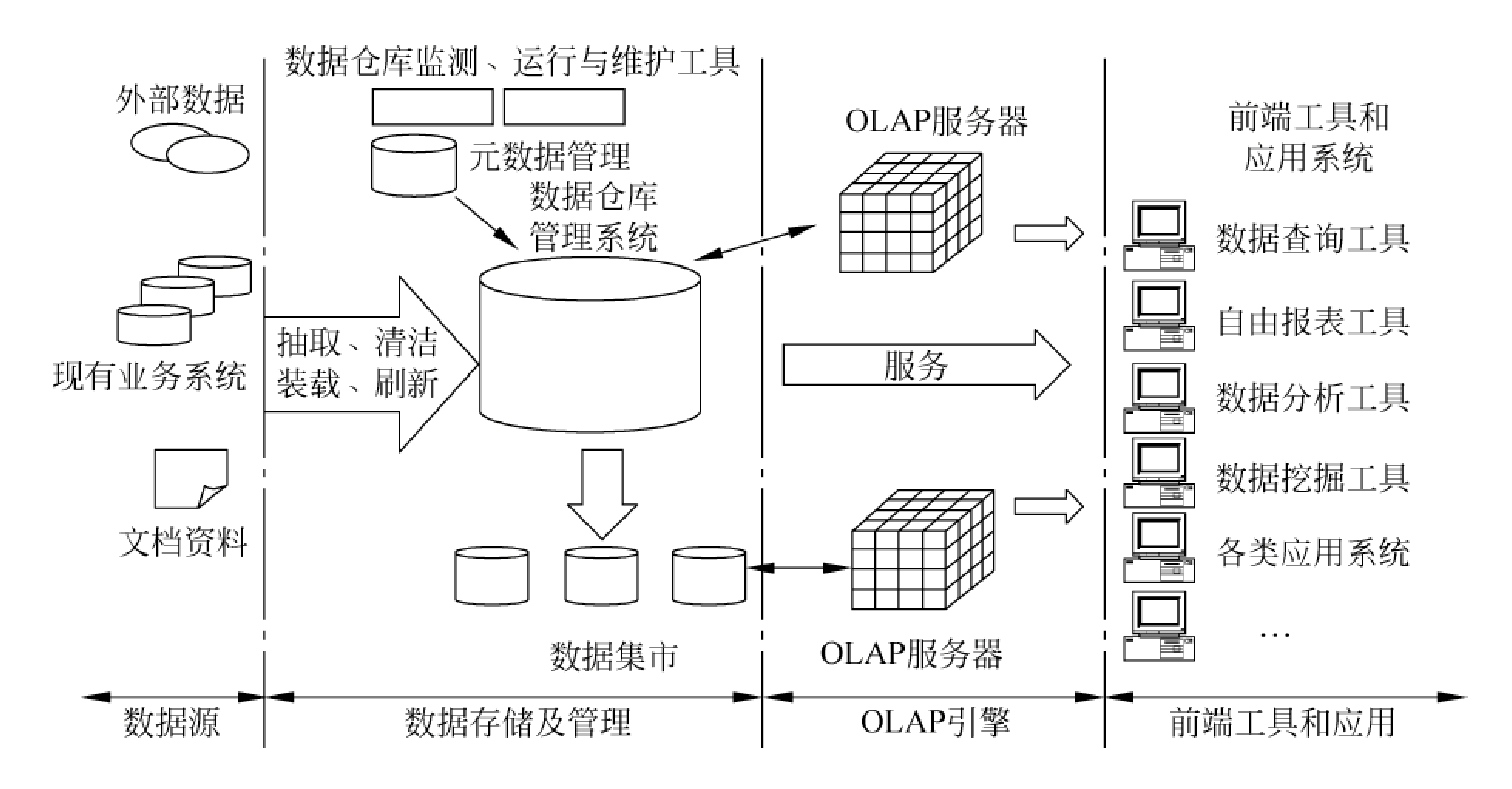


图 2.2　典型的数据仓库体系结构

(3) OLAP 引擎对分析需要的数据进行有效地集成，按照多维模型进行组织，实现多角度和多层次的分析，并预测趋势。按其具体实现可以分为 ROLAP(Relational OLAP)、MOLAP(Multi-Dimensional OLAP)和 HOLAP(Hybrid OLAP)。ROLAP 的基本数据和聚合数据均存放在 RDBMS 中；MOLAP 的基本数据和聚合数据均存放在多维数据库中；HOLAP 的基本数据存放在 RDBMS 中，聚合数据存放在多维数据库中。

(4) 前端工具和应用主要包括各种报表工具、查询工具、数据分析工具、数据挖掘工具以及各种基于数据仓库或数据集市的应用开发工具。其中数据分析工具主要针对 OLAP 服务器，报表工具和数据挖掘工具主要针对数据仓库。

2.1.3　数据模型设计

数据仓库的数据模型设计是构建数据仓库的关键，正确、完备的数据模型是用户业务需求的体现，是数据仓库成功与否最重要的技术因素。

由于数据仓库自身的特点，其数据模型的设计过程和传统操作型数据库数据模型的设计有很多不同，数据仓库的数据模型设计过程如图 2.3 所示。

不同设计阶段处理的主要问题可概括为：

- 企业模型——企业模型的建立是数据仓库数据模型设计的基础，以提高模型的可扩展性。
- 概念模型——概念模型用于完成数据仓库主题的确定，同时确定主题的范围。
- 逻辑模型——逻辑模型确定数据仓库的数据模式，主要关注大数据量的存储策略以及数据仓库的处理性能。
- 物理模型——物理模型使用具体的 DBMS 功能，进一步解决数据仓库的性能。

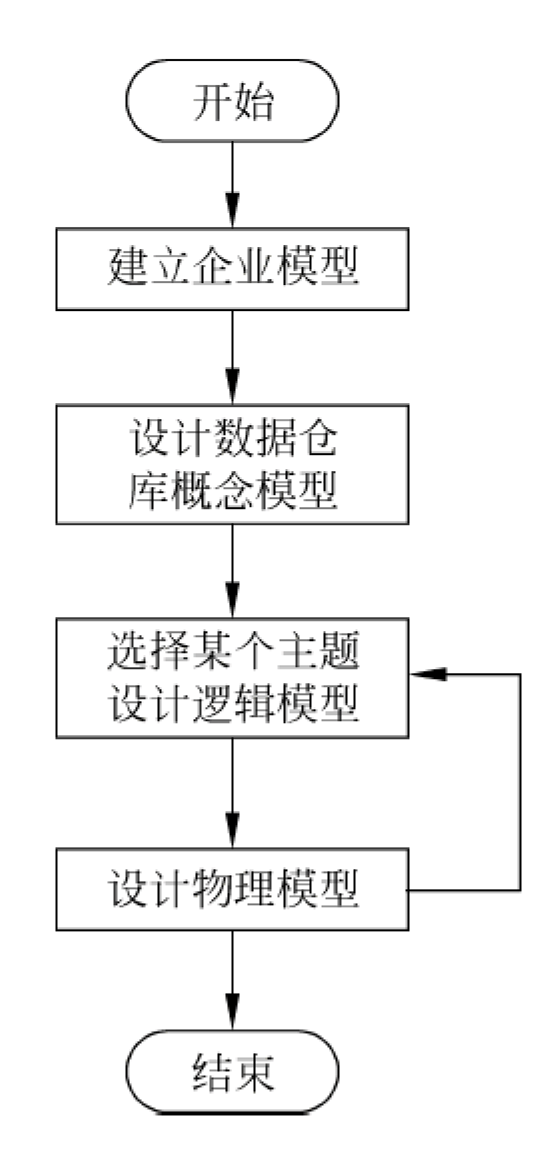


图 2.3　数据仓库的数据模型设计过程

1. 概念模型设计

数据仓库的概念模型描述了从客观世界到主观认识的映射。通过概念模型设计,可以确定数据仓库的主要主题及相互关系,它主要是依据建立的企业模型确定数据仓库的各个主题,主题来源于企业模型中的实体,主题的确定需要由最终用户和数据仓库设计人员共同完成。数据仓库的主题确定后,就可以根据主题将企业模型划分成不同的部分,同时将这种划分映射到相应的数据库模型,作为下一步逻辑模型设计的基础。

概念模型设计的主要步骤是:

(1) 确定主题

(2) 划定主题边界

概念模型设计是在原有的业务数据库的基础上建立一个较为稳固的概念模型。因为数据仓库是对现有数据库系统的数据进行集成和重组而形成的数据集合,所以数据仓库的概念模型设计,首先要对现有数据库系统进行分析和理解,了解现有数据库系统中有什么、怎样组织以及如何分布等,然后再考虑应当如何建立数据仓库系统的概念模型。一方面,通过现有数据库的设计文档以及在数据字典中的数据库关系模式,可以对企业现有数据库的内容有一个完整而清晰的认识;另一方面,数据仓库的概念模型是面向整个企业的,它为集成来自各个面向应用的数据库的数据提供了统一的概念视图。

概念模型设计是在较高抽象层次上的设计,因此设计概念模型时不用考虑具体技术条件的限制。

2. 逻辑模型设计

数据仓库的逻辑模型描述了数据仓库主题的逻辑实现,相对于关系数据库而言即是描述每个主题对应的关系表中关系模式的定义。

1) 模型选择

逻辑模型设计是数据仓库设计的重要步骤之一,因为它能直接反映出业务部门的需求,同时对系统的物理实施具有重要的指导作用。目前,数据仓库中较常用的逻辑模型是第三范式(Third Normal Form,3NF)和多维模型。以 Inmon 为代表的观点认为数据仓库建模应该采用基于传统的实体—关系,而以 Kimball 为代表的观点则认为数据仓库应该采用多维模型。通常在多维模型中以星型模式(Star Schema)最具代表性,所以有的学者把多维度模型直接称作星型模式。

(1) 第三范式

实体-关系又称为第三范式(Third Normal Form,3NF)是大多数传统数据库系统的建模方法。在数据仓库的逻辑模型设计中采用第三范式,具有非常严格的数学定义。如果从其表达的含义来看,一个符合第三范式的关系必须具备以下三个条件:

① 每个属性的值唯一,不具有多义性;

② 每个非主属性必须完全依赖于整个主键,而非主键的一部分;

③ 每个非主属性不能依赖于其他关系中的属性,否则这一属性应该归到其他关系中。

第三范式的定义基本上是围绕主键与非主属性之间的关系而做出的。如果只满足第一个条件,则称为第一范式;如果满足前两个条件,则称为第二范式,以此类推。因此,各级范式是向下兼容的。Inmon 提倡的第三范式建模与操作型数据库系统的第三范式建模在侧重点上有些不同。Inmon 提倡的数据仓库建模方法分为三层,第一层是实体关系层,即企业的

业务数据模型层，在这一层上和企业的操作型数据库系统建模方法是相同的；第二层是数据项集层，这一层的建模方法根据数据的产生频率及访问频率等因素与企业的操作型数据库系统的建模方法产生了不同；第三层物理层是第二层的具体实现。

(2) 多维模型

多维模型是一种面向用户需求的、容易理解的、访问效率高的设计方法。将数据仓库的数据组织成多维模型主要是基于数据仓库支持的大部分是OLAP应用，而OLAP要求数据按照多维模型的形式组织，以支持OLAP的钻取、切片和旋转等操作。同时多维模型与其他的数据组织形式相比，对数据进行了大量的预汇总操作以提高数据的查询速度，这对于既要处理大量数据同时又要保证用户查询效率的数据仓库系统更为适用。另外，以多维模型的形式组织数据也符合用户的查询习惯。

多维模型中数据是按照多维的形式组织的，维度是用户观察数据的角度，如时间、地域等。以用户的一次通话为例，其中包括了时间、话务类型、通达地市等多个观察的维度。维度是有层次的，如时间维可以按照"年—月—日"的层次划分，维度的不同层次决定了所展示数据的详细程度。根据维度的不同特性可以将维度分类，不同的分类将影响维的实现方式。

根据维度层次结构的特点，可以将维度划分为：

• 均衡的层次结构

均衡的层次结构中，每个层次的所有分支都降至同一级别，而且每个成员的逻辑父代是上一级成员。例如，移动业务中的通话区域维，可以划分为"省—地市—交换机"三个层次，如图2.4所示。

• 非均衡的层次结构

非均衡的层次结构中，每个层次的分支降至不同的级别，但是同一成员的逻辑子代全部位于同一级别，如图2.5所示。

其中，客户类型层次中的两个成员个人客户和集团客户分别降至不同的级别。

• 不整齐的层次结构

不整齐的层次结构中，每个层次的分支降至不同的级别，并且同一成员的逻辑子代位于不同的级别，如图2.6所示。

其中，费用类型层次中的成员通话费的逻辑子代分别降至费用类型细分和最小费项两个不同的级别。

根据维度生成方式的不同，可以将维度划分为：

• 业务实体

维度来源于业务系统中的各个业务实体，它们是与主题关联的实体，例如客户、产品和销售渠道等。

• 事实属性

维度也可以是来源于事实的属性，它们是对事实的分类，如电信业务中通话行为这一事实中的通话类型(长途、本地通话)就是通话行为的属性。一般这样的维在实体关系模型中表现为关联实体的属性。

• 业务实体属性

这部分维度是与主题关联的各个业务实体的属性，是对业务实体的分类，如客户属性中的性别、工作性质。

福建
南平
南平GSM交换机
其他
莆田
莆田GSM交换机
莆田2号华为交换机
其他
宁德
宁德GSM交换机
宁德2号交换机...
其他
厦门
厦门GSM交换机2
厦门GSM交换机
其他
泉州
泉州GSM交换机4...
泉州GSM交换机2
泉州GSM交换机3
泉州GSM交换机1
其他
漳州
漳州2号华为交换机
漳州GSM交换机
其他
龙岩
龙岩GSM交换机
其他
福州
福州GSM交换机3
福州GSM交换机2
福州1号交换机...
其他
三明
三明GSM交换机
其他

图 2.4　均衡的维度层次结构

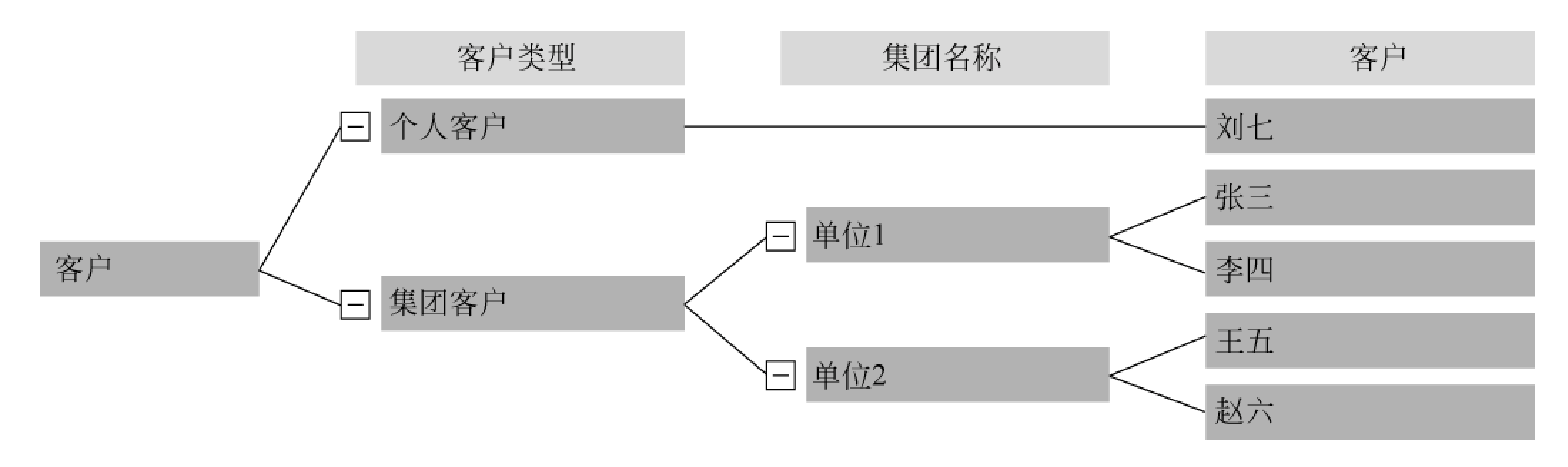


图 2.5　非均衡的维度层次结构

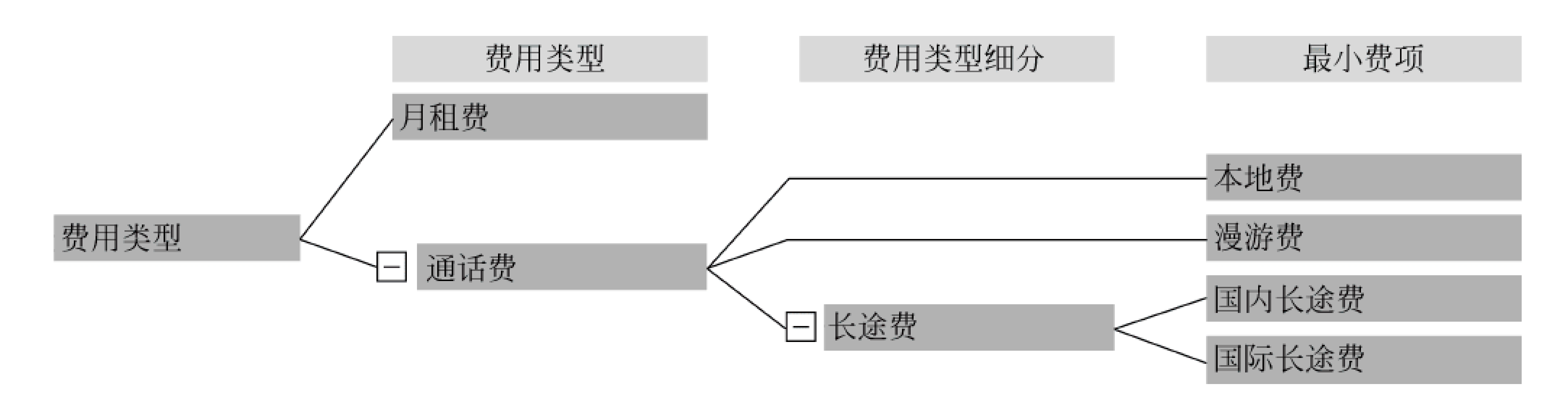


图 2.6　不整齐的维度层次结构

• 派生维

派生维是基于事实表中其他维或度量生成的逻辑维，如电信业务中的付费方式(预付费，后付费)维是通过套餐维生成的。增加派生维度，不会增加事实表中的数据量。派生维的生成依赖于用户的业务定义。

除了维度的概念外，多维模型中的另一个重要概念是度量。度量是从现实系统中抽象出来的，描述数据的实际含义，如电信企业发展的用户数、出账费用等。度量一般都是数值类型，通过使用聚集函数得到。通过度量可以对所观察的事物进行一定的评价。

根据度量对数据集的聚集方式不同，度量可以划分为：

• 分布的

设数据被划分为 n 个集合，函数在每一部分上计算得到一个汇总值。如果将函数用于 n 个汇总值得到的结果，与将函数用于所有数据得到的结果一样则该度量是分布的。例如，在电信业务中通话费用这个度量就是分布的，这种形式的度量所应用的聚集函数一般是 count()、sum 等。

• 函数的

如果聚集函数可以由一个具有 M 个参数的函数表示，且每个参数都可以用一个分布汇总函数求得，则该度量是函数的。如 avg()可以由 sum()/count()计算得到，其中 sum()和 count()是分布汇总函数，例如月平均使用次数就是一个函数的度量。

• 整体的

如果一个聚集函数无法用具有 M 个参数的函数表示，则该度量是整体的，如 rand()、count(distinct)等，例如使用业务的用户数就是一个整体的度量。

不同的维度和度量是通过事实联接起来的。事实是对某类事件或某种状态的记录，例如，一次通话或者业务受理等，事实一般都包括若干度量提供对事件的评价，例如通话这一事实包括通话时长、通话费用等度量。也有一部分事实不包括任何度量，它们反映了某类事件的发生，如“某人在某天被停机”。

Kimball 提倡的数据仓库的多维模型，一般也称为星型模式，有时也加入一些雪花型模式。星型模式是为了将数据分割为执行起来容易理解的格式而设计的，它是一种多维的关系，由一个事实表(Fact Table)和一组维表(Dimension Table)组成。每个维表都有一个维作为主键，该主键链接到事实表，所有这些维组合成事实表的主键。事实表的非主属性称为事实(Fact)，一般都是数值或其他可以进行计算的数据，而维大都是时间、地域等。

• 事实表。每个数据仓库或数据集市都包括一个或多个事实表。星型模式或雪花型

模式的中心是一个事实表。通常,事实表包含大量的行,有时当事实表包含企业一年或几年的历史数据时,可能有数亿条记录。事实表的主要特点是包含数值数据(事实),而这些数值数据可以汇总以提供有关企业运营历史的信息。每个事实表还包含一个由多个部分组成的索引,该索引包含作为外键的相关维表的主键,而维表包含事实记录的特征。事实表不应包含描述信息,也不应包含数值度量字段,以及使事实与维表中的对应项相关的索引字段之外的任何数据。

- 维表。包含描述事实表中的事实记录的特征。有些特征提供描述性信息,有些特性用于指定如何汇总事实表数据以便为分析者提供有用的信息。维表包含有助于汇总数据的特性的层次结构。

数据仓库的负载主要有两种:一种是回答重复性的问题;另一种是回答交互性的问题。对于以第一种负载为主的部门数据集市,当数据量不大、报表较固定时可采用多维模式;对于中央数据仓库,考虑到系统的可扩展性、投资成本和易于管理等诸多因素,可采用第三范式。

2)模型比较

逻辑模型的设计既可以采用星型模式或者雪花型模式,也可以采用第三范式。

在数据库逻辑模型设计中有一个规范化的过程,以减少数据冗余。范式是衡量数据库规范化程度或者深度的一种方法,具有非常严格的数学定义。根据数据规范化程度的不同,由低至高分为第一范式、第二范式、第三范式、Boyce-Codd 范式、第四范式和第五范式。对于一般的数据库系统而言,只做到第三范式,如果一个数据模型满足第三范式,就可以认为该模型的冗余度已经很低了。

在数据仓库逻辑模型设计中,采用第三范式可以达到以下目的:

- 减少数据冗余,减少数据的存储要求。
- 便于数据抽取。采用第三范式,减少了单表中的数据冗余,因此在进行数据装载时要比星型模式和雪花型模式使用较少的关联操作,数据的装载速度较快,同时减少数据冗余也降低了数据之间的依赖关系,便于数据的并行抽取。例如电信企业数据仓库数据模型设计时,如果在欠费表中加入用户套餐等冗余信息后,在抽取欠费数据时必须首先等到用户数据和欠费数据同时准备好后才能开始抽取数据。
- 便于重用业务系统的数据模型,提高开发速度。

但是满足第三范式的数据模型在查询时,特别是针对 OLAP 应用的查询,需要进行的数据关联操作较多,查询效率低。而星型模式和雪花型模式通过对事实表的预先汇总和引入冗余字段,提高了查询效率。但是进行数据汇总的操作会丢失一部分信息,因此数据汇总方式的选择应根据用户的查询需求,如果需求发生变化特别是需要增加新的维度时,则需要修改模型,甚至有时要对数据进行重新装载。

数据仓库的数据粒度可以分为历史细节级、当前细节级、轻度综合级和高度综合级四个层次,其中轻度综合级和高度综合级的数据都是根据用户需求进行一定程度的汇总,因此对轻度综合级和高度综合级的数据易于使用星型模式或雪花型模式,以提高数据的访问速度。而对于当前细节级数据可以采用满足第三范式的数据模型以提供各种细节信息。

采用多维模型设计时需要注意的问题如下:

• 事实表的设计

事实表主要来自业务系统中的操作记录，例如电信业务中的通话、缴费和出账等，同时也有一部分事实表是对主题状态的记录，如用户停开机状态的记录。

在设计事实表时，应对事实表中包含的度量进行分类，如果该度量是一个整体度量，则需要对该度量生成不同汇总级别的事实表以保证数据的准确性。例如在通话行为中，如果有一个度量是通话用户数，它是一个整体度量，因此要对该度量生成不同汇总层次的事实表。在设计事实表中应尽量避免使用整体度量。

此外，还应注意将具有相同维度的事实表通过加入多个度量进行合并，以减少数据仓库中表的数量和数据冗余，但是在进行事实表合并的同时应兼顾数据的抽取效率，因为合并事实表常会导致数据抽取时需进行表之间的关联，例如，如果将用户的欠费数据和销账数据合并为同一个事实表，则需要将欠费数据和销账数据按照用户标识进行关联形成一条记录后装载到事实表，这样将影响数据的抽取效率。一般对高度汇总级的数据进行事实表的合并。

• 维表的设计

维表主要来自于业务系统的各个实体以及实体的属性，同时还有一部分是根据用户需求生成的派生维。

一般不使用业务系统的主键作为维表的主键，维表的主键一般选择无意义的顺序的数字作为主键。这主要是为了保证对维表中数据变化的处理，同时也增加了维度层次组合的灵活性。由于业务系统都是通过主键进行关联的，因此增加新的维表主键会导致在数据抽取时需要对数据进行转换，影响数据抽取效率。同时维表中应保存业务系统中的主键以标识业务系统同一意义的数据。

一般地，业务实体单独作为维表，不同实体之间的关联通过主、外键实现。对于用户定义的派生维通常需加入时间字段以保存用户定义的变更。

3）逻辑模型设计

数据仓库的逻辑模型可以认为是数据仓库开发者和使用者之间就数据仓库的开发进行交流和讨论的工具与平台，同时对系统的物理实施具有重要的指导作用，通过实体和关系勾勒出整个企业的数据蓝图。

逻辑模型设计主要包括如下几个方面：

（1）主题域分析

在概念模型设计中，确定了几个基本的主题域。但是，数据仓库的设计是一个逐步求精的过程。一般是一次一个主题或一次若干个主题逐步完成。所以必须对概念模型设计步骤中确定的几个基本主题域进行分析，选择首先要实施的主题域。选择主题域所要考虑的是它应足够大，以构建一个可应用的系统；它还要足够小，便于较快地开发和实施。

（2）数据粒度层次划分

数据仓库逻辑设计的一个重要问题是数据仓库的粒度层次划分，其适当与否直接影响数据仓库的数据量和所适合的查询类型。一般需要将数据划分为详细数据、轻度综合、高度综合三级或更多级的粒度。粒度层次的划分是由数据的行数决定的，数据行数越多，所需存储空间越大，粒度划分的级别就越多。

（3）数据分割

数据分割是提高数据仓库性能的重要手段，它把逻辑上是统一整体的数据分割成较小

的、可以独立管理的物理单元进行存储,以便于重构、重组和恢复,提高创建索引和顺序扫描的效率。确定分割策略主要是指选择适当的数据分割标准,一般应考虑以下几方面因素:数据量、数据分析处理的要求、简捷性以及粒度划分策略等。其中,数据量的大小是决定是否进行数据分割和如何分割的主要因素;数据分析处理的要求是选择数据分割标准的一个主要依据,因为数据分割与数据分析处理的对象紧密联系,设计者还需考虑到所选择的数据分割标准应是自然的、易于实施的,同时也应考虑数据分割标准与粒度层次划分是相适应的。

(4) 关系模式定义

数据仓库的每个主题都是由多个表实现的,这些表之间以主题的公共键码关联在一起,形成一个完整的主题。在概念模型设计时,确定了数据仓库的基本主题,并对每个主题的公共键码、基本内容等进行描述,即对选定的当前实施的主题进行模式划分,形成多个表,并定义各个表的关系模式。

4) 数据粒度

数据粒度是指数据仓库中保存数据的细化或综合程度的级别。细化程度越高,粒度越小;反之,细化程度越低,粒度则越大。数据粒度的确定对于数据仓库数据模型的设计非常重要,主要原因在于:

- 数据仓库保存业务系统的历史数据,随着时间的推移,数据不断增加,数据仓库保存了大量数据。在有限的直接存储设备上保存这些信息,同时又能够较快地响应用户查询,如果仅保存细节数据,将减少数据的在线保留周期,而且数据查询的效率也会降低。
- 用户的分析型操作大部分是对汇总信息的查询、对数据趋势的观察,如果仅保存细节数据,则完成分析型操作时需要遍历大量数据,对系统的软硬件要求都很高。

实际上,数据仓库的数据是按照不同粒度保存的。确定数据粒度旨在延长有效数据的在线时间,提高查询效率。

确定数据粒度时,需要考虑以下问题:

(1) 确定是否使用多重粒度

单一粒度和多重粒度的主要区别是在数据的冗余存储上,而不是指是否对数据进行汇总,单一粒度和多重粒度都存在汇总数据。单一粒度是指在数据进入数据仓库时先保存为细节数据,当数据的保留周期到期时,对细节数据进行汇总形成综合数据,同时将细节数据导出到其他慢速存储设备上,单一粒度中细节数据和汇总数据之间逻辑上没有重叠;多重粒度是指数据在进入数据仓库时,同时以细节数据和汇总数据存在,当数据的保留周期到期时,将细节数据导出。多重粒度中细节数据和汇总数据有一部分是重叠的。对于细节数据量较大的数据仓库而言,如果以单一粒度存储,对近期数据的查询效率较低。因此,对这类数据仓库建议采用多重粒度存储。

确定是否需要使用多重粒度,需要对数据仓库保存的数据量进行估算,W. H. Inmon 给出了数据量估算公式。

对每个已知表 i 计算:

- 估算一行所占字节数的最大值 L_i 和最小值 l_i。
- 统计一年内可能出现的数据行数的最大值 M_i 和最小值 m_i。

- 统计五年内可能出现的数据行数的最大值 N_i 和最小值 n_i。
- 计算每个表键码的字节数 K_i。

一年内数据量估算为：

$$最大值 = \sum (L_i + K_i) \times M_i$$

$$最小值 = \sum (l_i + K_i) \times m_i$$

五年内数据量估计为：

$$最大值 = \sum (L_i + K_i) \times N_i$$

$$最小值 = \sum (l_i + K_i) \times n_i$$

根据估算的数据量，参照表 2.1，确定是否使用多重粒度。

表 2.1　不同数据量级与数据粒度的对应关系

1 年内数据量(行)	5 年内数据量(行)	数据粒度策略
10 000	100 000	设计简单
100 000	1 000 000	采用单一数据粒度
1 000 000	10 000 000	最好采用双重数据粒度
10 000 000	20 000 000	必须采用双重数据粒度

除了数据量外，是否使用多重粒度还应考虑细节数据的保留周期。如果细节数据的保留周期较长，可考虑使用多重粒度以保证数据的查询效率。值得注意的是不同行业、不同用户需求可能导致不同的细节数据保留周期。

(2) 确定粒度级别

确定粒度级别是指确定数据仓库中汇总数据的汇总程度，数据越详细，数据粒度越低。如果数据汇总程度较低，则需要大量的资源处理数据；如果数据汇总程度较高，则会降低查询的灵活性。

如前所述，数据仓库的数据粒度层次分成历史细节级、当前细节级、轻度综合级和高度综合级。一般而言，历史细节级和当前细节级保存的都是最低粒度的数据。对粒度级别的选择主要针对轻度综合级和高度综合级而言，以便大部分查询可以在轻度综合级或高度综合级完成。

确定粒度级别，需要综合考虑用户需求和实际存储设备容量。众所周知，数据仓库的开发是一个反复的过程，因此数据粒度级别也需要和最终用户反复交流确定。如果用户对最终展示的数据要求更详细则要降低粒度；反之，如果用户对轻度综合数据的查询大部分是数据汇总操作，可以适当提高粒度级别，以提高用户查询速度。

确定粒度级别时需要注意的问题如下：

- 对数据进行汇总时一般应保留主题对应的公共键码，同一主题下的数据是通过主题的公共键码进行关联。
- 选择粒度级别时，考察汇总后生成的某些派生字段的汇总方式。例如在产品主题中对一天中产品的使用情况进行汇总，可以得到每天不同产品使用的用户数，但是用户数对时间而言不存在有规律的汇总方式(或者说是整体的)，对这种派生字段必须提供多个不同粒度的汇总数据。

(3)选择数据汇总方式

在确定数据粒度时要选择数据汇总方式,可供选用的数据汇总方式主要包括:

• 按维汇总

由上述的多维模型可知,维是用户观察数据的角度,对应关系模型而言就是表中的某个字段。维是有层次的,按维汇总就是指选取一个维度,将数据按照维的不同层次进行汇总,这是最常用的数据汇总方式。例如电信业务的详单数据中可以按时间维进行汇总,从而得到某天某个用户的通话总次数、总时长;再对一个月内的用户通话进行汇总,从而得到一个月内一个用户的通话总次数、总时长,进而再汇总到年。时间维是经常用来进行汇总的一个维度。

按维汇总时,还可以考虑用户观察数据的其他角度,加入其他维度。但是如果维度过多,将会影响数据的汇总效果。此时,可同时进行模式转换的汇总。例如,电信业务话单的通话级别维标志该条话单是本地还是长途通话,采用按照时间维的不同层次进行数据汇总,如果加入的维度很多,可以删除通话级别维,加入本地通话次数、长途通话次数字段。

• 提取数据子集

提取数据子集是根据用户的需求,因为用户可能只对一个数据集合中某类数据子集感兴趣,因此可以通过数据过滤,只保留部分数据供用户查询和分析,从而降低存取的数据量。例如,在电信业务的话单中,用户只对入网一个月以内的用户通话数据感兴趣,可以提取这部分细节数据供用户分析使用,其他数据通过汇总保存在较高粒度级别上。

• 变更数据模型

通过改变数据模型对数据进行汇总,将某些分类信息转换成字段存储。例如电信用户账单中保存的是每个月每个用户在不同费项上发生的费用,因此该表的主键是月份、用户标识和费项类型,这样数据行数较多,可以通过改变数据模型,即删除费项类型,加入不同费项类型的费用字段,这样表的主键是月份和用户标识,数据行数明显减少。

• 建立广义索引

广义索引是指对数据集合中的数据特征进行统计,这样用户仅需要查看统计结果就可以了解数据的大致情况,统计方法包括最大最小值、数据平均值和排序等。例如针对电信业务的欠费,可以建立本月欠费前 100 名的用户以方便查询。

建立广义索引是基于用户的查询需求,通过了解用户经常需要回答的问题,可以为用户建立相应的广义索引,提高查询效率。

数据仓库中,数据粒度之所以重要,是因为它深深地影响数据仓库数据量的大小,以及数据仓库所能回答的查询类型。实际中,需要在数据仓库的数据量和查询的详细程度之间进行权衡。

如图 2.7 所示为两个粒度不同的数据仓库,其中左图是低粒度的数据仓库,每次通话都被详细记录下来,到月底每个客户平均有 200 条记录(当月每个电话都记录一次),因此共需 40 000 个字节;右图是高粒度的数据仓库,数据代表每个客户一个月的综合信息,每个客户一个月只有一条记录,这样的记录大约只需 200 个字节。当提高数据粒度级别时,数据仓库所能回答查询的能力会随之降低。换言之,在一个很低的粒度级别上可以回答任何问题,但在高粒度级别上所能回答的问题有限。

针对图 2.7 提出的问题“上星期某某客户是否给某某人打了电话?”,在低粒度级别上,

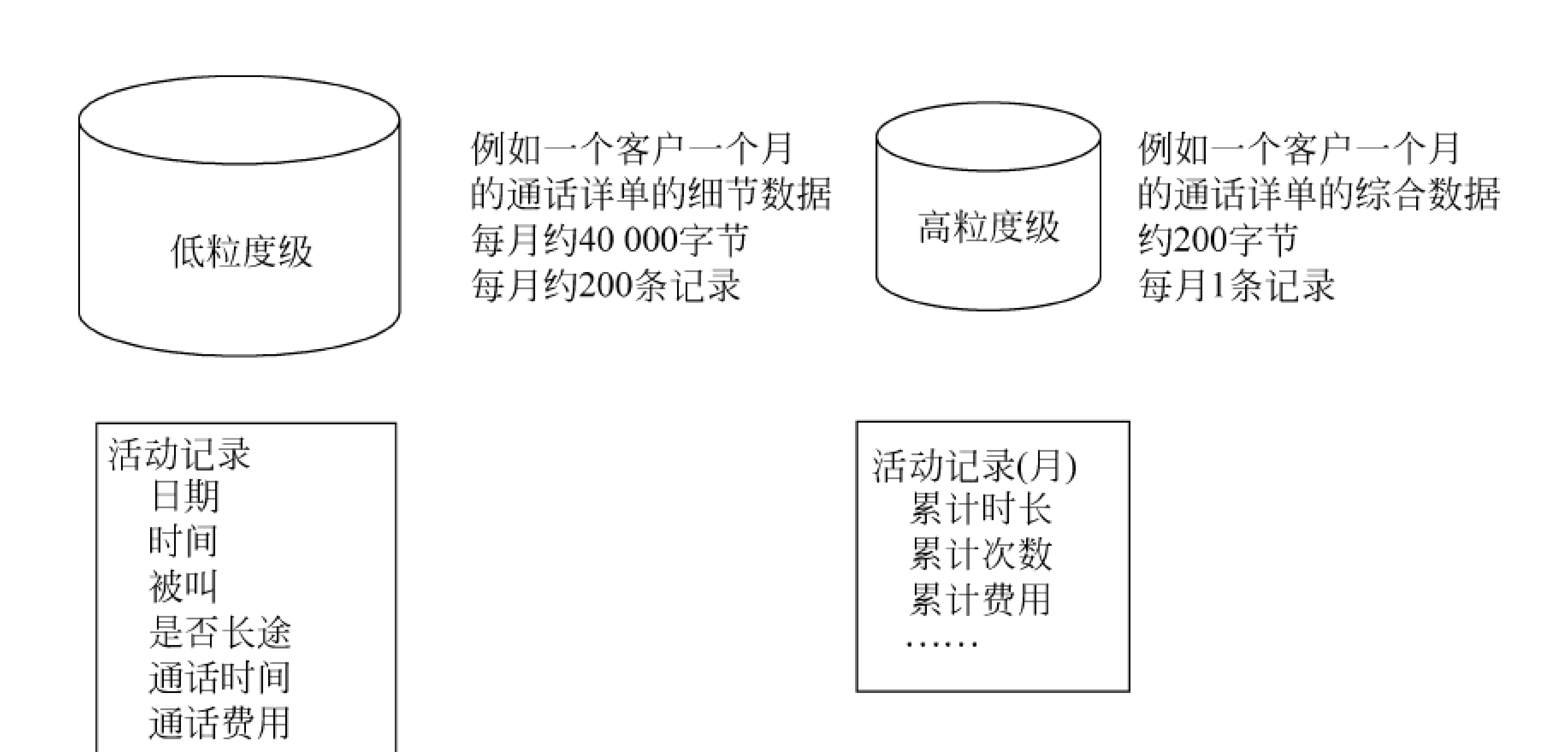


图 2.7　不同粒度的数据仓库

这个问题是可以回答的，虽然这种回答将花费大量开销去查询大量的记录，但查询结果是确定的。然而，在高粒度级别上则无法明确地回答这一问题。因为在数据仓库中存放的只是所拨打电话的总数，无法确定其中是否有一个电话是打给某人的。

但是，DSS 很少对单个事件进行检查，通常是针对某种数据集合进行处理，这意味着需要查询大量记录。例如"上个月客户从某地呼出的长途电话平均多少个?"，对 DSS 而言这类查询是很常见的。当然，它既可以在高粒度级别也可以在低粒度级别上得到回答。值得注意的是，在不同的粒度级别上所使用的资源具有相当大的差异。在低粒度级别上回答这一问题需要查询每条记录，所以需要大量的资源。但在高粒度级别上，数据进行了很大的压缩。

显然，如果数据仓库的存储空间很有限，高粒度级别要比低粒度级别的效率高得多。

高粒度级的数据仓库不仅占用的存储空间少得多，而且需要的索引项较少。然而不仅要考虑数据量和存储空间的问题，为了访问海量数据处理能力同样也是一个应考虑的要素，所以数据仓库中数据压缩非常有用。数据压缩后会大大节省所占用的存储空间，节省所需的索引项以及处理器资源。

综上所述，对于数据粒度的权衡，在数据仓库设计时必须仔细加以考虑。

5）数据分割

数据粒度是通过对数据的汇总来减少数据量，提高数据的存储和查询效率。但是并不能解决同一粒度下由于数据量过大导致的数据查询效率较低的问题。对于这类问题需要使用数据分割技术。数据分割是指将数据分散到各个物理单元中以便能够独立处理，提高数据处理效率。数据分割后的数据单元称为分片，分片数据没有交叉。

数据分割的意义在于：

- 提高查询速度

通过分割，降低查询的数据量，同时将大表分割成若干小表，易于建立表索引。

- 便于数据重组

数据重组是指将数据按照一定的规则进行新的组合，例如将电信业务话单表分割成每月一张表，用户可能按照一定的规则将月份分成几个组，这样当查询某个组的数据时只要扫

描该组中的月份对应的表,数据量相对较小。

• 便于表维护

通过数据分割,降低单表的数据量,减轻数据维护的工作量。

• 增加系统并行性

数据分割可以增加数据抽取和查询的并行性。数据仓库中,数据分割的关键问题不是该不该分割而是如何分割。数据分割的主要工作是选择分割的标准,数据分割没有固定的标准,需要根据用户的需求和数据使用方式共同确定数据分割标准。

选择分割标准时需要考虑的主要因素如下:

• 数据分割尽量均匀

数据分割旨在将数据量较大的数据分割成若干小的分片以提高查询速度,因此选择数据分割的标准应尽量保证数据在各个分片中的分布较均匀。例如,对通话详单可以按照时间进行分割,由于每天通话的次数相差不多,对某天通话进行查询时可以获得较高的查询速度。

• 用户的查询需求

在选择数据分割标准时应考虑用户的查询需求,以便将用户的大部分查询在一个分片内完成,达到提高系统性能的目的。例如,用户在查询账单时经常按照用户所属地市进行查询,因此对账单按照用户所属地域进行分割,这样用户大部分查询可以在一个地市的账单表中完成。但是,如果数据分割的标准改为按用户标识的末尾数字,则用户在查询一个地区的账单时需要扫描全部分片,这样查询效率要比不进行数据分割时还要低。因此,数据分割一般按照某个维度的一个层次的取值,常用的维是时间、地域和产品等。

• 数据汇总的方式

数据分割还应考虑数据汇总方式,如前所述,选择高粒度级别可减少数据量,同时提高用户某类查询的速度,因此在对低粒度级别数据进行分割时还应考虑数据汇总到高粒度时的汇总方式,一般选取的数据分割标准是高粒度级别中的某个维度。

• 易于数据重构

数据重构是数据分割的反操作。易于重构是指采用分割标准进行分割的数据易于组成整体以满足用户的查询需求。

数据分割标准是严格地由开发人员选择的。然而,在数据仓库中,日期几乎是分割标准中一个必然的组成部分。

数据分割可以在应用层也可以在系统层。系统层的数据分割一般是依赖 DBMS 或者操作系统提供的功能,不同的 DBMS 提供的数据分割方法也不同。而应用层的数据分割主要是按照一定的规则,将数据分割为若干逻辑小表。在逻辑模型设计时考虑的数据分割是应用层的。

数据仓库开发人员面临的主要问题之一是在系统层还是在应用层进行数据分割。在系统层进行数据分割在一定程度上是 DBMS 和操作系统提供的功能。在应用层进行数据分割则是由应用程序代码完成的,这是由开发者和程序员严格控制的。当在应用层进行数据分割时,DBMS 和系统无法知道一种分割与另一种分割之间的关系。

通常,在应用层进行数据分割是很有意义的。最为重要的是在应用层上每年的数据可以有不同的定义。不同年份的数据定义,可以相同也可以不相同(例如不同版本)。在系统

层进行数据分割时，DBMS 不可避免地希望只有一种数据定义。假如数据仓库中保存的数据时间较长(如达到十年)，并且数据定义经常变化，让 DBMS 或操作系统去管理一个本该只有一种数据定义的系统将是毫无意义的。

在应用层进行数据分割的另一重要特点是它能从一个处理集转移到另一个处理集而没有损失。在数据仓库环境下，当负载和数据量成为真正的负担时，这一特点则是一种真正的优势。

对于数据分割而言，最严峻的挑战是能否在分割中加入索引而不会明显地影响其他操作。如果一种索引可随意加入，则分割是十分理想的；否则，分割还需更精细些。

数据分割的主要实现手段包括重构、自由索引、顺序扫描(可选的)、重组、恢复和监控等。

如果数据粒度和数据分割进行了很好的设计，几乎所有数据仓库设计和实现的其他问题都将迎刃而解；否则难以真正实现。

6) 数据划分

数据划分与数据分割不同，数据分割对关系模型而言是指对元组的划分，而数据划分是指对实体的属性按照一定的分组原则划分为不同的属性组，即对数据模式的分解。一般地，数据划分针对主题对应的实体以及各个与主题相关的业务实体。

逻辑模型设计中进行数据划分的主要原因在于：

- 从数据存储角度来看，实体中有的实体属性更新较为频繁，而其他属性更新频率较低。由于数据仓库要保存业务系统的变更历史，因而对于每次属性的变化，数据仓库一般采用新建记录的方式进行记录，这样如果将变化频率不同的字段放在同一张表中，将浪费较多的存储空间。
- 从数据访问角度来看，表中的部分字段是经常被访问的，而其他字段访问次数较少，如果将所有字段存放在一张表中，则表会增大，影响数据访问速度。

基于上述原因，在逻辑模型设计时需考虑对表进行划分，采用的划分标准主要是按照字段变化频率和按照业务规则。

(1) 按照字段变化频率

根据字段的变化频率，将表中的字段分成“比较稳定”、“有时变化”和“经常变化”三类，相应地将表拆分成变更频率不同的三个子表，例如电信用户资料表的各字段按照稳定性划分如图 2.8 所示。

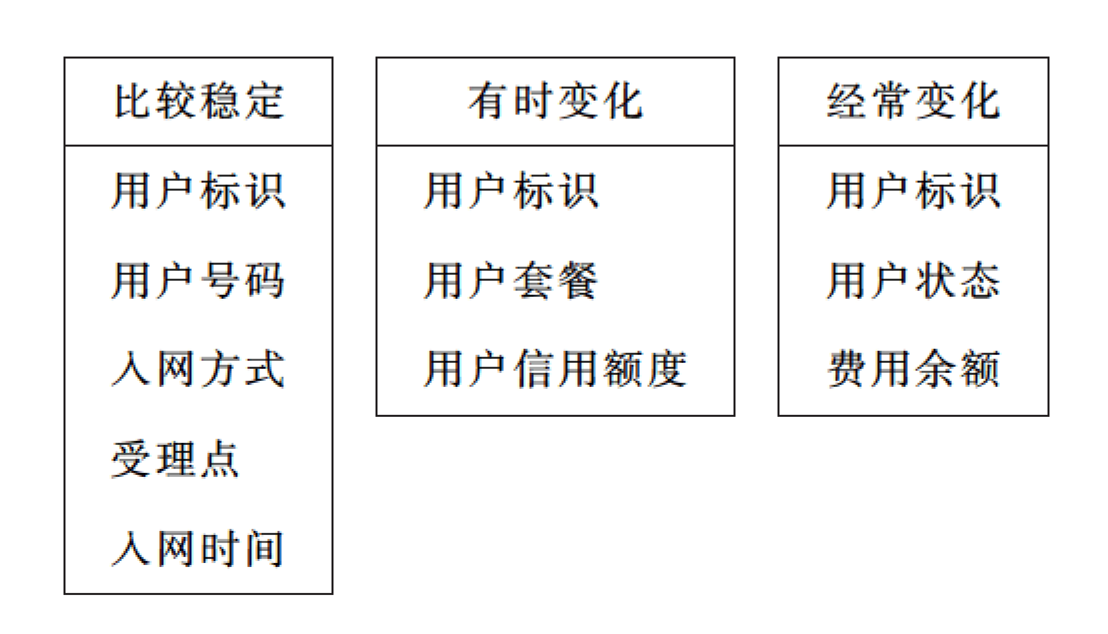


图 2.8 按照字段变化频率划分电信用户资料表的各字段

按照字段变化频率对表进行划分后,数据的存储空间占用较少。

(2) 按照业务规则

按照业务规则是指按照业务系统的业务规则将字段分成共有数据和专有数据。W. H. Inmon 提出了数据项集合(Data Item Set,DIS)方法对表进行划分。DIS 方法将数据分成初始数据、二次数据、联接数据以及各种数据类型。其中,初始数据是指主题中仅出现一次的属性,例如客户标识、入网时间等;二次数据是指关联实体的属性,例如产品名称等,一般是一些描述性字段;联接数据是指本主题与其他实体间的关联,例如客户和产品等。数据类型是指按照业务规则对实体进行分类,可以看成是类的继承关系,例如客户按照消费程度可以分为大客户和普通客户;按照所处的状态可以分为停机客户、正常客户和离网客户等。可以对实体按照不同的标准进行分类,不同分类之间是组合的关系,如图 2.9 所示。

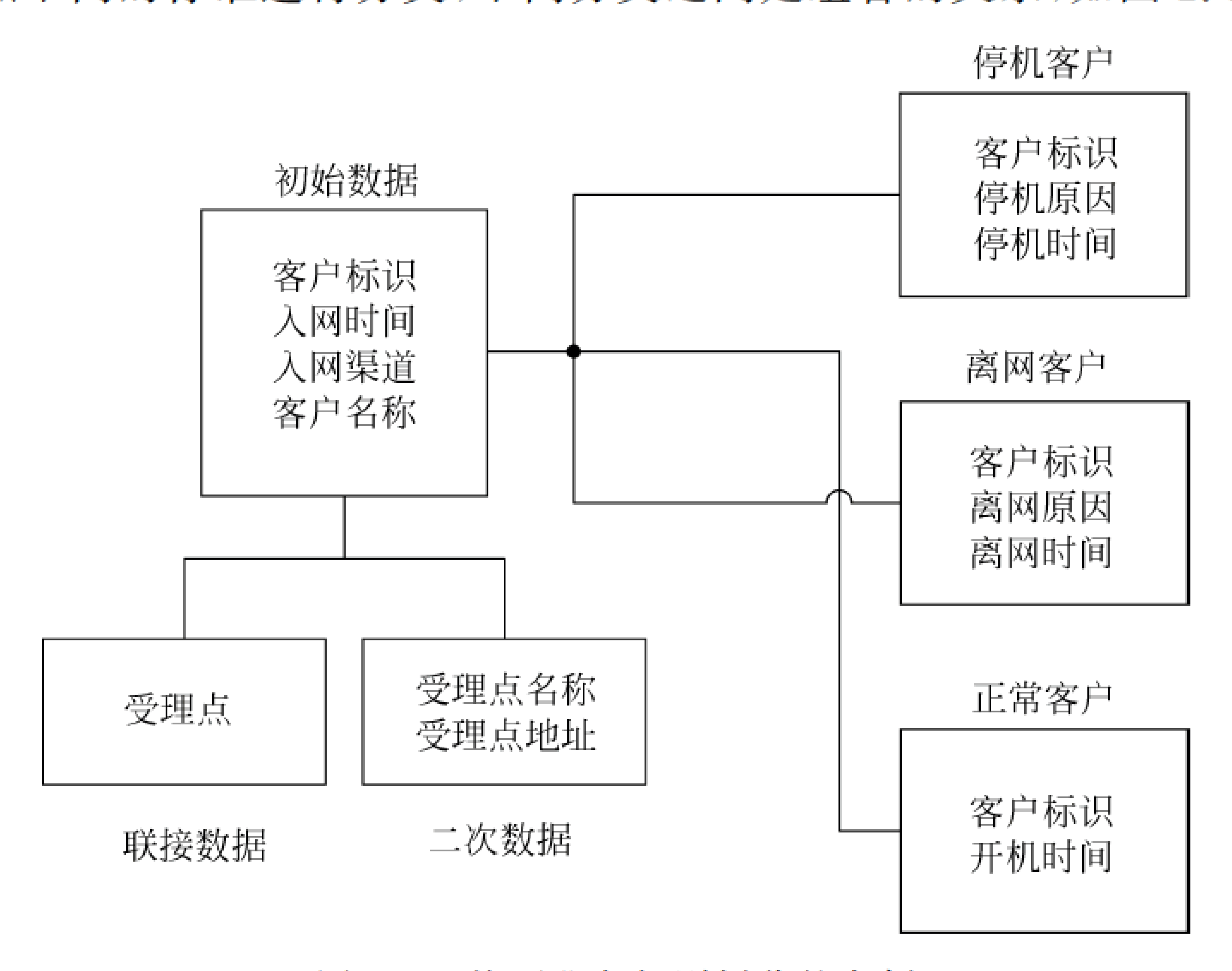


图 2.9 按照业务规则划分的实例

按照业务规则对数据进行划分,有利于数据访问。

7) 数据模式

数据仓库的数据模式包括星型、雪花型和星型-雪花型,三者都是以事实表为中心,不同之处只是外围的维表之间的关系不同而已。

(1) 星型

星型模式的每个维度都对应一个唯一的维表,维的层次关系全部通过维表中的字段实现,所有与某个事实有关的维度,都通过该维度对应的维表直接与事实表关联,所有维表的主键组合起来作为事实表的主键,如图 2.10 所示。星型模式的维表只与事实表发生关联,维表与维表之间没有任何关联,具体示例如图 2.11 和图 2.12 所示。

在图 2.11 中,地域是一个与销售事实表关联的维度,地域维的层次是“省—地市”,这一层次关系由维表中的省代码和地市代码字段实现。

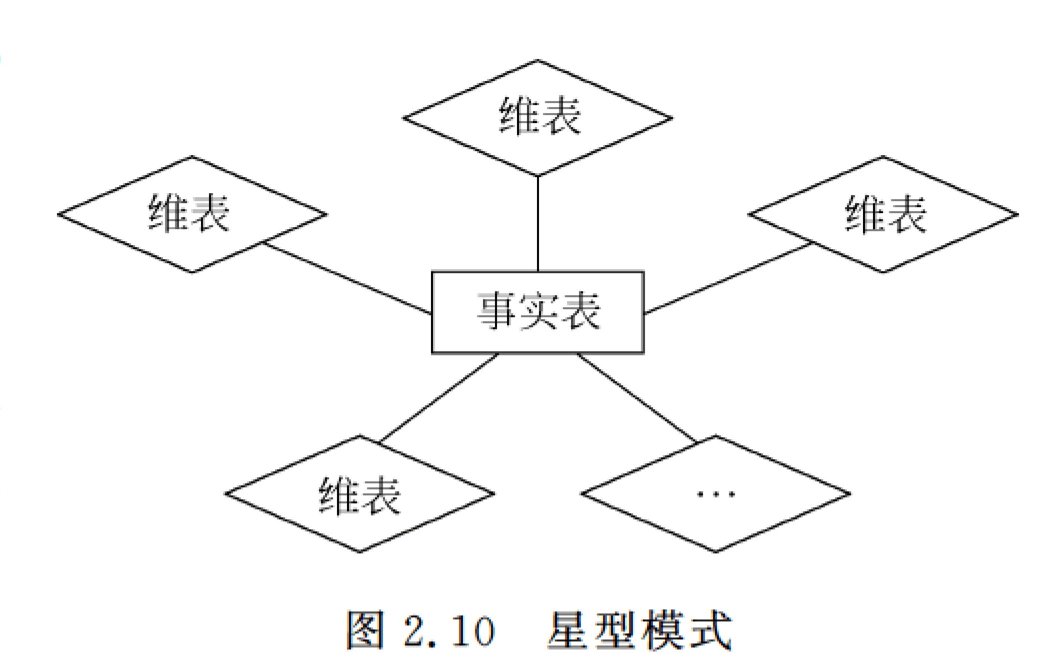


图 2.10 星型模式

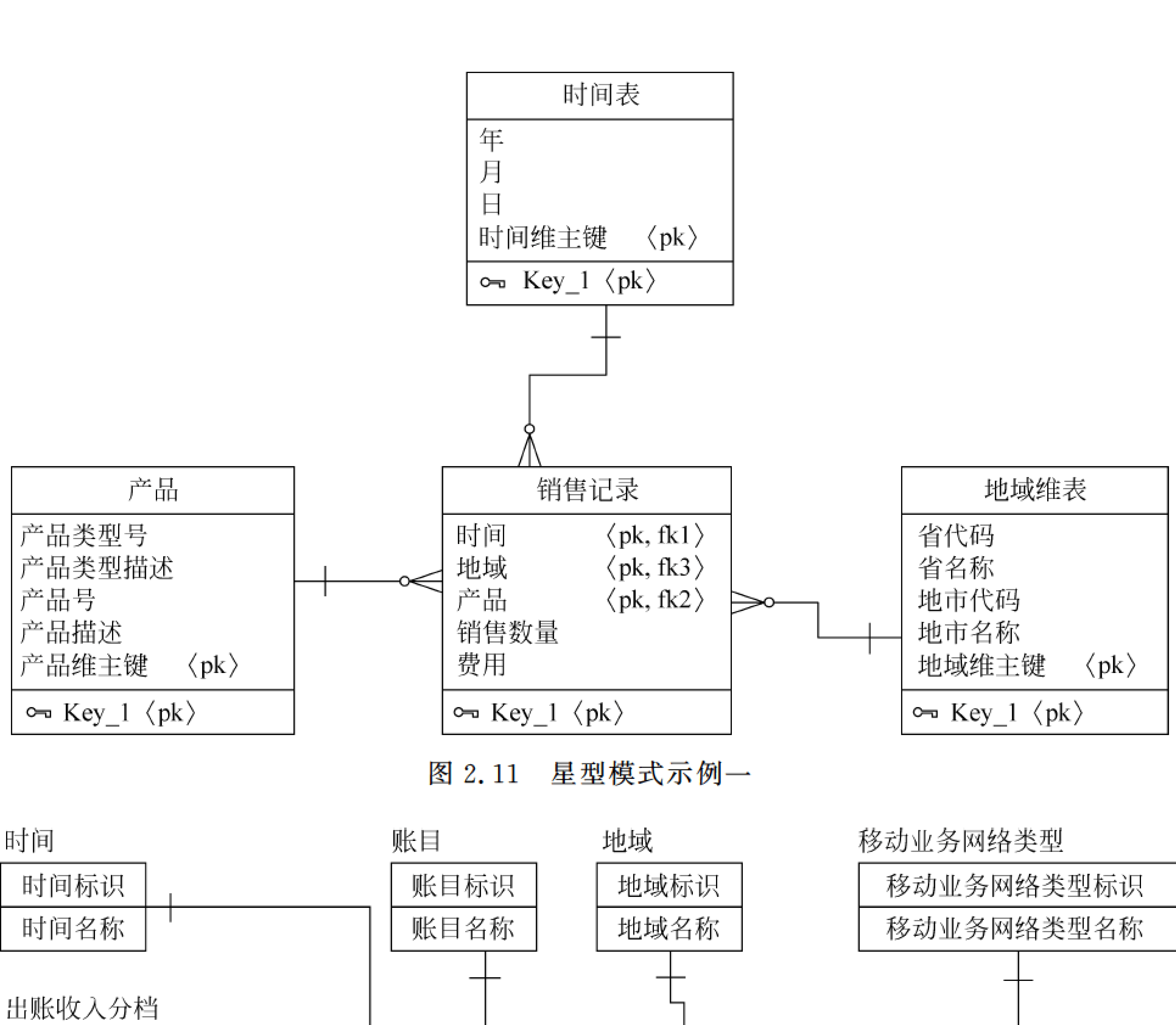


图 2.11 星型模式示例一

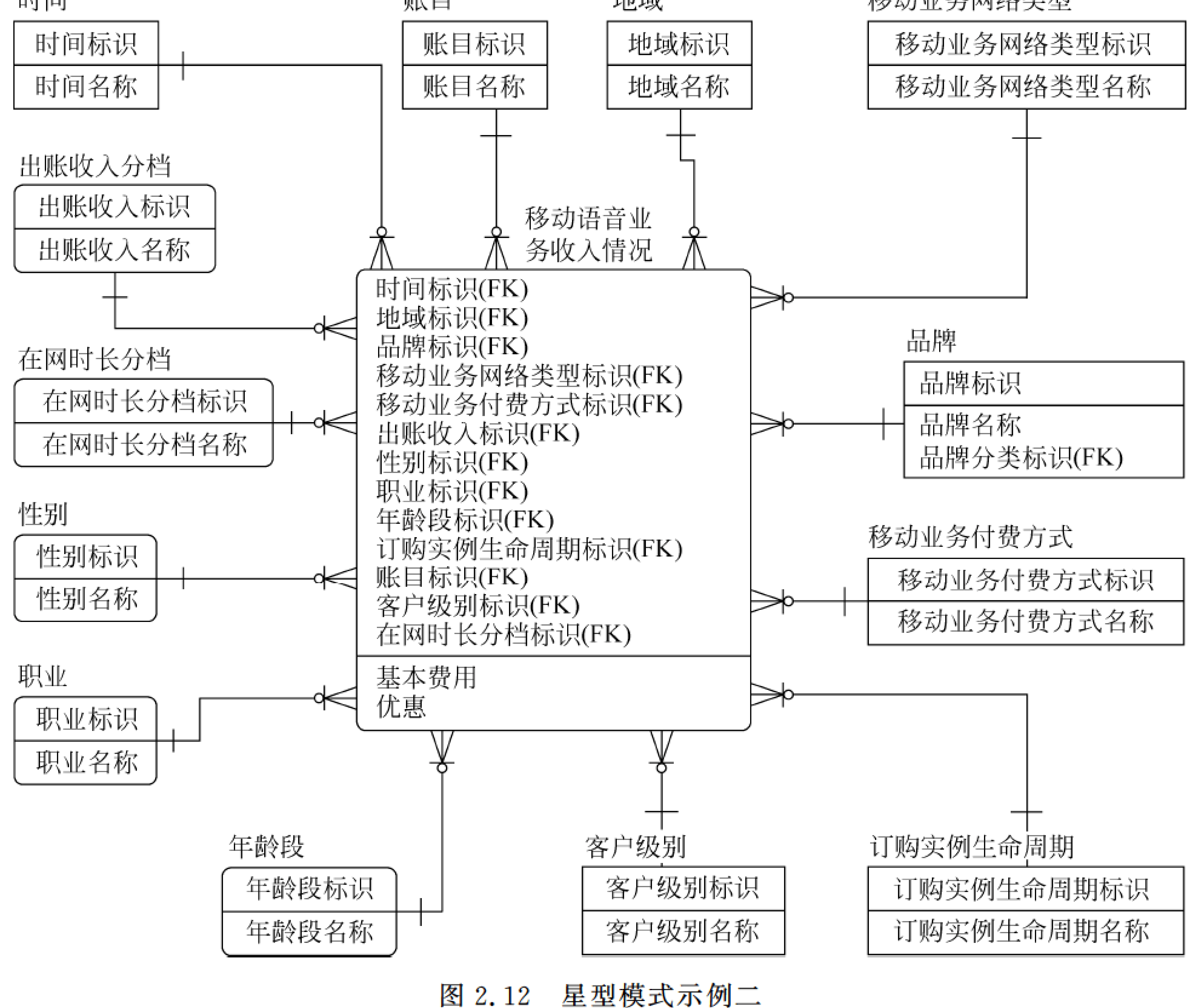


图 2.12 星型模式示例二

星型模式的特点是：

• 维表的非规范化

星型模式中，维表保存该维度的所有层次信息，因此是非规范化的，这样减少了查询时

数据关联的次数,提高查询的效率。但是由于维表保存了所有的层次信息,使得维表之间的数据共用性较差,例如电信业务中基站和受理点两个维的层次关系分别是“地市—区县—基站”和“地市—区县—受理点”,这两个维中都有地市和区县,但是由于所有的层次信息都保存在各自的维表中,因此地市和区县分别保存在两个维表中,同一信息之间的统一是通过人工维护的。

• 事实表的非规范化

星型模式中,所有维表都直接和事实表关联,因此事实表也是非规范化的,这样减少了查询时数据关联的次数,提高查询的效率。但是这种方式也限制了事实表中关联维表的数量,如果关联的维表数量过多将会造成大量数据冗余,同时对事实表进行索引也很困难。

• 维表和事实表之间的关系

星型模式中,维表中的主键在事实表中作为外键存在,因此维表和事实表的关系是一对多或一对一。如果维表和事实表之间是多对多的关系时,则不能直接采用星型模式,必须通过对维表或者事实表的处理,如对维表中的成员组合进行编码或者在事实表中加入新的字段,但这都要求成员的组合数量固定,如果数量不固定,同时维表的数据量又很大的情况下,星型模式的实现较为困难。

(2) 雪花型

事实上,维表只与事实表关联是正规化的结果。如果将经常合并在一起使用的维度进行正规化,即所谓的雪花型模式,如图 2.13 所示。同星型模式相比,最大区别是雪花型模式对维表进行了规范化,即用不同维表之间的关联实现维的层次,具体示例如图 2.14 所示。

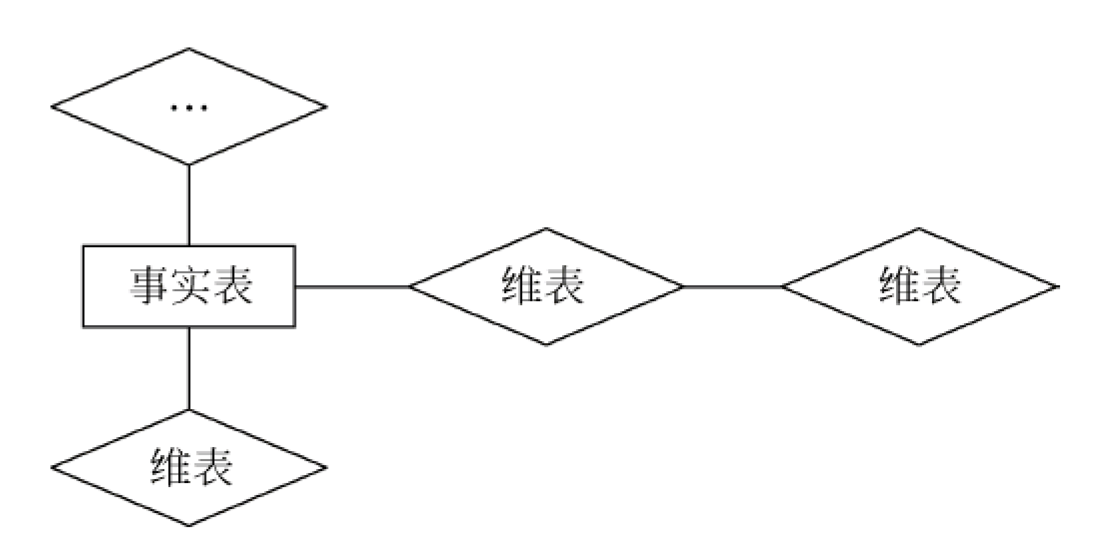


图 2.13 雪花型模式

图 2.15 所示为对于上述基站和受理点的例子使用雪花型模式加以实现。

雪花型模式的特点是:

• 通过维表的规范化实现维表重用,简化维护工作。但是,查询时使用雪花型模式要比星型模式进行更多的关联操作,反而降低了查询效率。
• 雪花型模式中有些维表并不直接和事实表关联,而是与其他维表关联,特别是对于派生维和实体属性对应的维而言,这样就减少了事实表中的一条记录。因此在维度较多特别是派生维和实体属性较多的情况下,雪花型模式较为合适。但是当按派生维和实体属性维进行查询时,首先进行维表之间的关联,然后再与事实表关联,因此查询效率低于星型模式。
• 使用雪花型模式可以实现维表和事实表之间多对多的关系,例如在上述例子中加入销售人员维,如果一次销售可由多个员工完成,采用雪花型模式实现如图 2.16 所示。

综上所述,星型模式结构简单,查询效率高。而雪花型模式通过维表的规范化,增加了

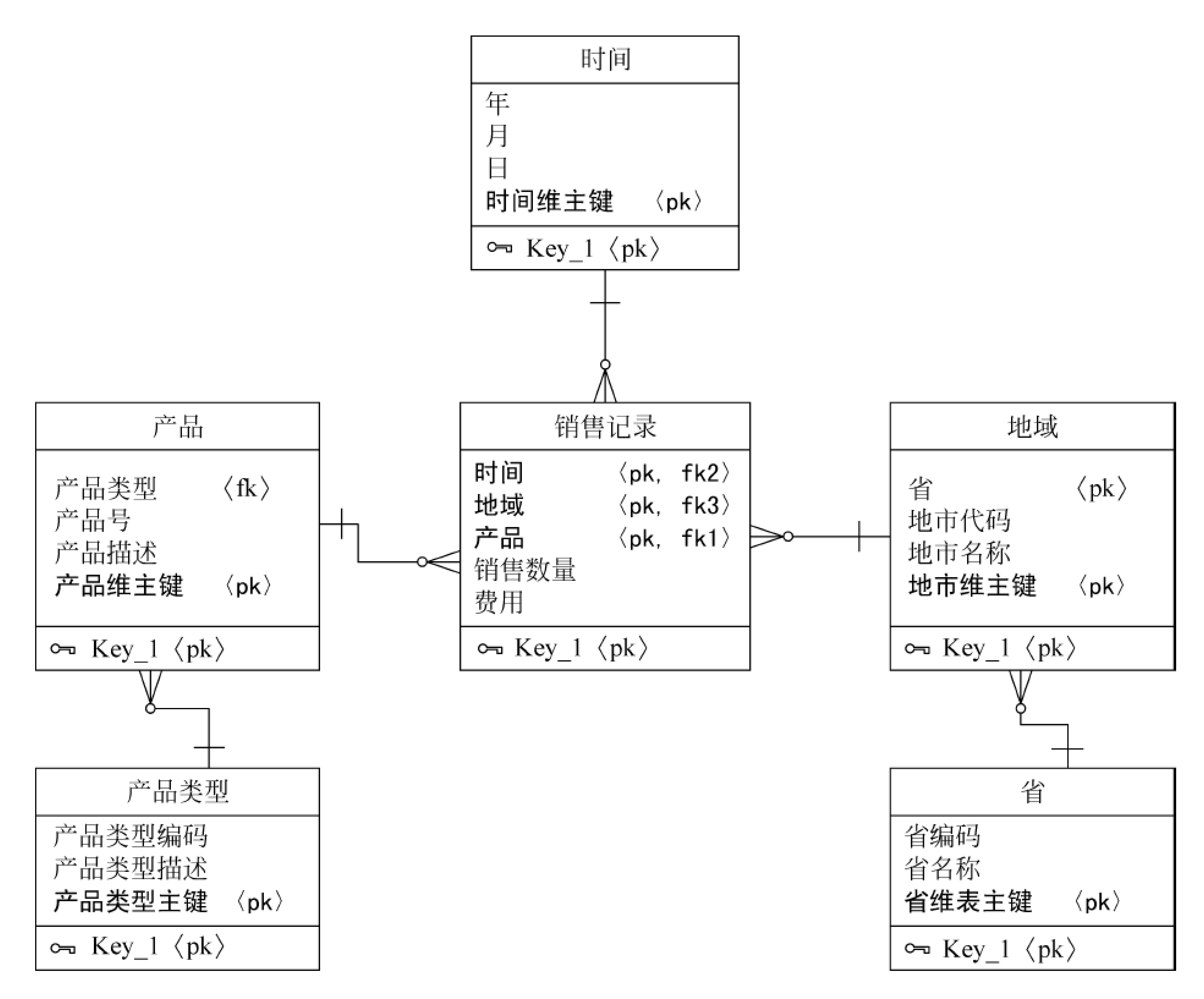


图 2.14 雪花型模式的示例

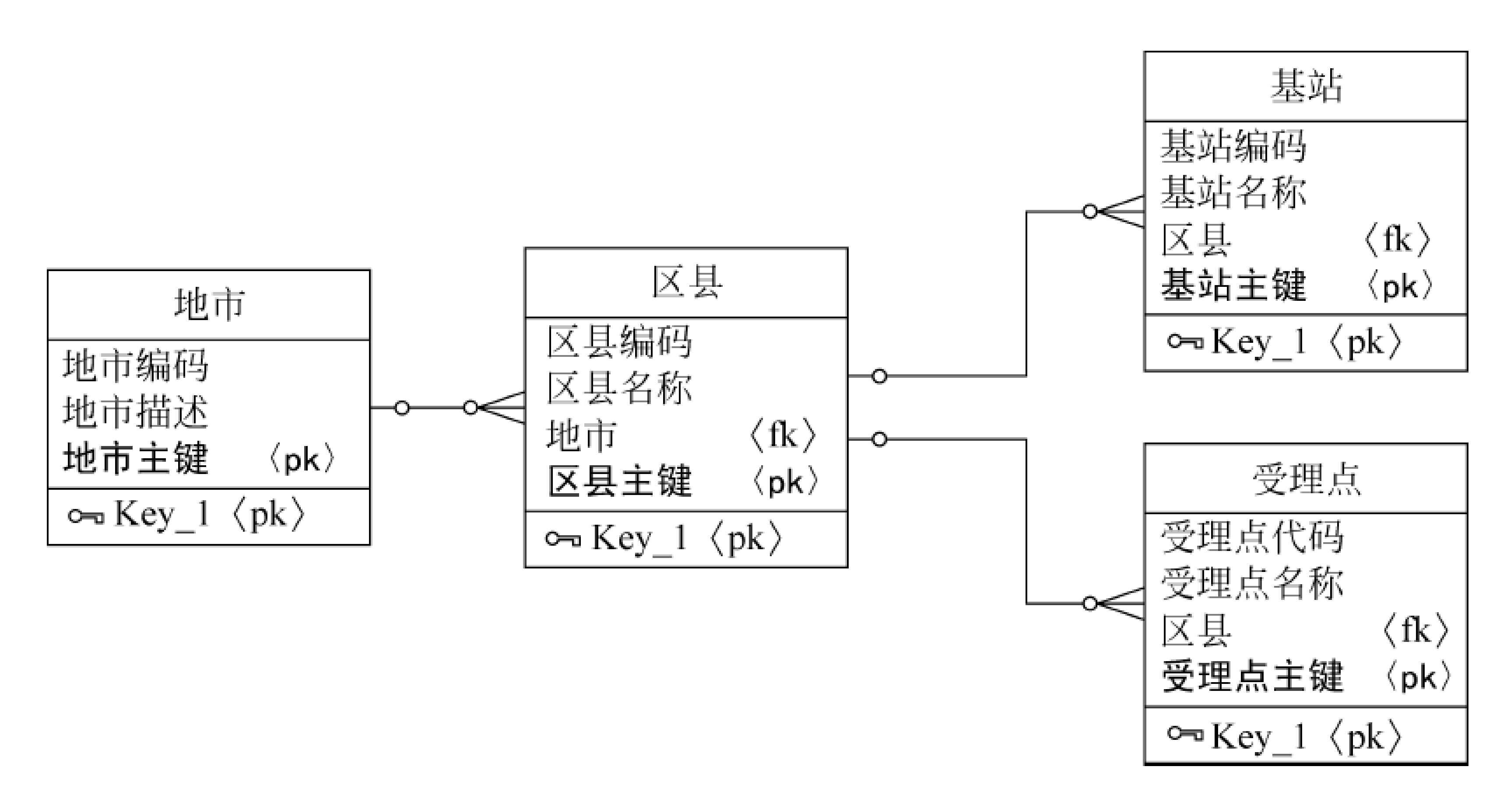


图 2.15 雪花型模式的维表共用

维表的共用性。实际应用中，经常将星型和雪花型模式综合起来，即使用星型模式的同时将其中的一部分维表进行规范化，提取一些公共的维表，这样既保证了较高的查询效率，又简化了维表的维护。

(3) 星型-雪花型模式

星型-雪花型模式是星型和雪花型模式的结合，打破了星型模式只有一个事实表的限制，且这些事实表共享全部或部分已有维表，如图 2.17 所示。

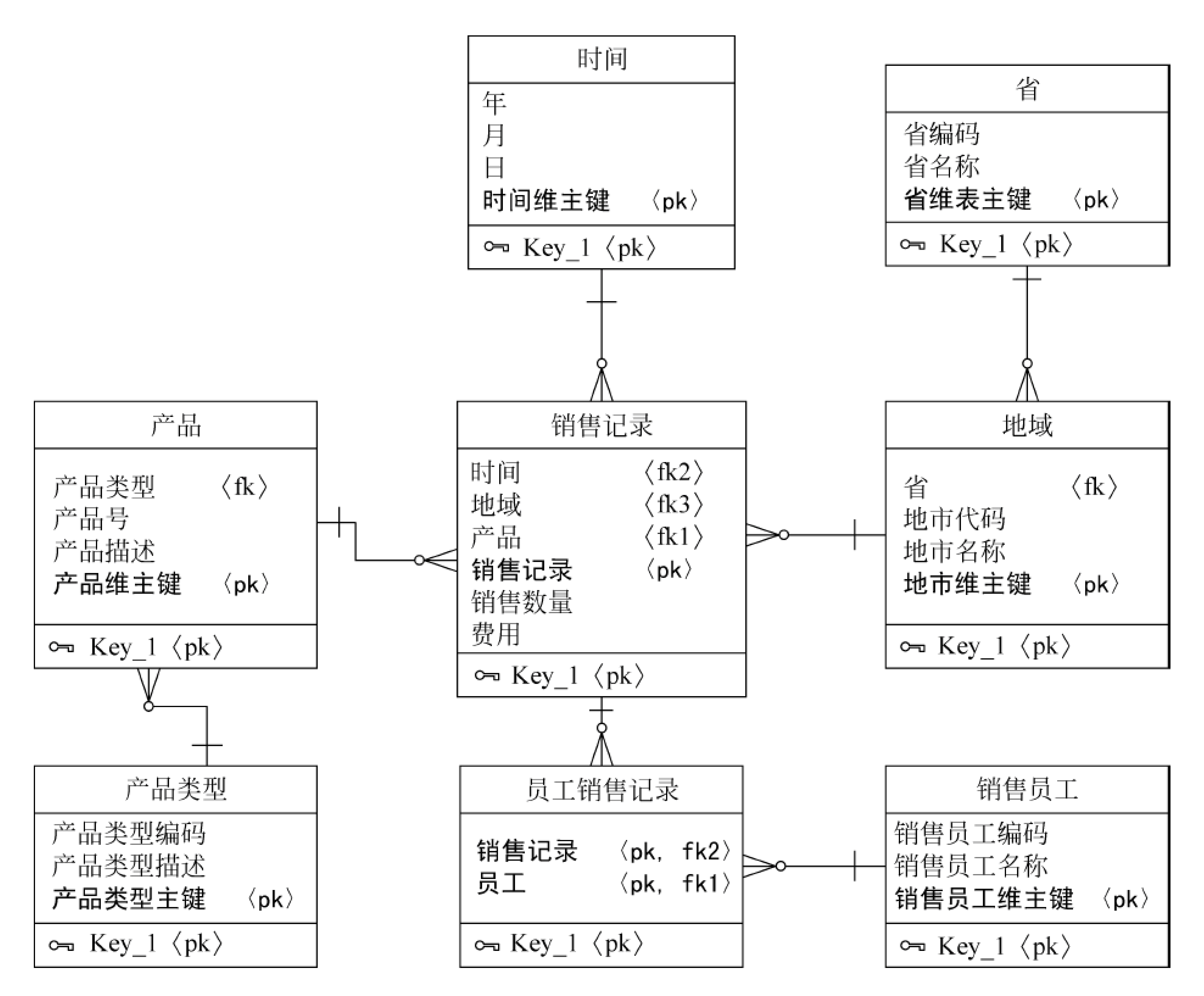


图 2.16 雪花型模式实现多对多关系

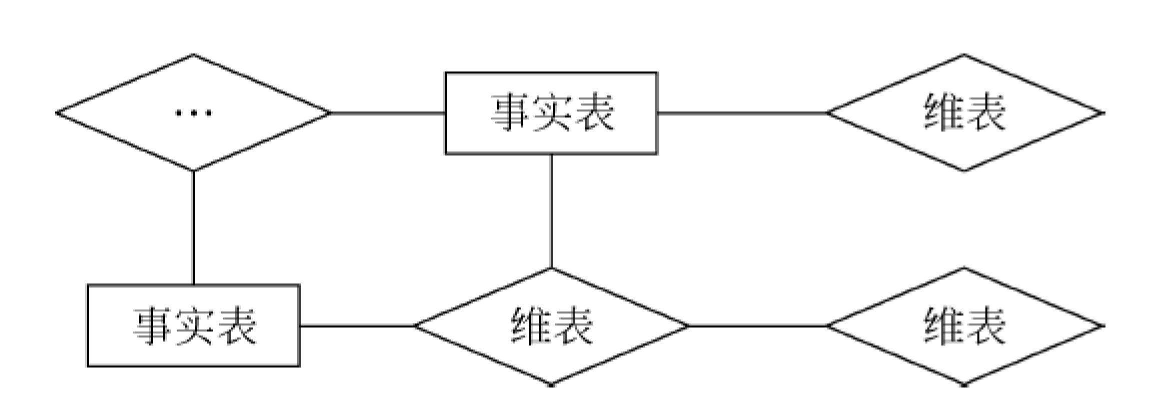


图 2.17 星型-雪花型模式

3. 物理模型设计

数据仓库的物理模型设计是指在逻辑模型的基础上，使用 DBMS 提供的功能对逻辑模型进行物理上的优化，即根据数据仓库的特点和性能要求，将逻辑模型转化成数据在物理设备上的存储结构与存取方法的过程，如表结构、索引、数据约束、数据类型和数据格式设计等，由项目经理和数据仓库架构师共同完成。

具体地，物理模型设计主要包括如下几个方面。

1）数据存储结构设计

数据存储结构设计主要包括数据文件存储设计、数据文件存储分配以及数据缓冲大小等，其目标是提高系统的 I/O 能力。

作为数据仓库的基础，DBMS 往往可以向用户提供多种存储结构，每种存储结构各有其独特的实现方式。在利用 DBMS 构建数据仓库时，应当统一考虑数据存储时间、存储空间效率、数据维护成本等各方面因素，选用合适的存储结构。

2）数据索引策略设计

数据索引策略设计是指根据数据的特征，如数据的基数和数据的使用情况选择适当的索引，例如使用 Sybase 数据库时，对低基数的数据可以建立位图(Bitmap)索引。

数据仓库中数据存储量十分庞大，远远超过一般的数据库系统。当数据仓库数据的更新频率不高，往往采用定期导入新数据的方法。因此，为了有效提高数据仓库的运行效率，可对常见查询所访问的数据进行分析的基础上，设计较为复杂的索引策略。由于数据仓库的数据具有非易失性，因此尽管索引的设计与建立工作量较大，却易于维护。

3）软硬件配置确定

数据仓库与其他业务系统不同，尤其需要对数据容量进行估算，这是由于数据仓库是对以往历史数据的集成，如果设计时不加以考虑，很快就会造成灾难性后果。数据仓库的容量估算应该是可预见的，首先确定核心明细数据的存储年限，相关表的平均字段长度值×每年的记录数×(每年预计的增长)，然后再加上 20％的冗余，以及磁盘预留的 20％的冗余，即可以得到数据仓库的数据量估算值。

数据仓库的处理能力不仅与容量有关，还与具体的关系数据库的性能息息相关。如何在 Oracle、SQL Server、Sybase 甚至 MySQL 之间寻找平衡，既要考虑实际的预算，也要视实际需求而定。关于硬件的配置，既需要发挥软件的功能，满足实际的处理要求，也要为将来的系统扩展保留一定空间。

4）数据存储策略设计

由于同一个主题的数据可以存放在不同的介质上，为了提高存取效率，设计者常常按照数据的重要程度、粒度、使用频率以及响应时间等要求将数据分别存放在不同的存储设备上。

数据存储策略包括表的归并、表的物理分割和表的预联接等。

(1) 表的归并

表的归并是指将用户经常要查询的放在一起，减少数据 I/O 次数，提高查询效率，表的归并是 DBMS 提供的功能。

(2) 表的物理分割

表的物理分割是指使用 DBMS 提供的功能，对逻辑模型中的数据进行再次的分离。通常，逻辑模型按照一定的业务规则对数据进行分割，而物理模型一般是按照数据的使用信息对数据进行分割，实际中将数据的逻辑分割和物理分割综合应用。例如，在逻辑模型中对话单表按照主叫号码的所属地域进行分表设计，在物理模型中对每个表再按照通话日期进一步分区。

(3) 表的预联接

表的预联接是指根据数据的使用情况，利用 DBMS 提供的功能(例如 Oracle 的实体化视图、Sybase 的联接索引等)，对经常关联的表事先进行预联接处理，以提高数据访问速度。

5）表的物理特性设计

表的物理特性设计是指对表的初始块大小等物理特性的设计，主要参考表中数据量的情况，通过对物理特性的设计提高数据生成效率。

6）存储分配参数确定

在创建传统数据库的过程中，需要确定数据块大小、缓冲区大小以及缓冲区数据等具体

与数据存储分配相关的参数。通常不同的数据库厂商都会根据其产品的应用实例给出推荐的配置参数,供设计人员参考进行初始配置,然后在系统维护过程中根据实际情况(数据的增长速度、用户查询的数据量和频率)进行调整。数据仓库是建立在 DBMS 上的,因此两者在这一点上相同。

2.1.4 ETL 设计

除了数据模型设计外,数据仓库设计还包括数据装载接口设计。数据模型设计即前面提及的数据仓库概念模型设计、逻辑模型设计和物理模型设计;数据装载接口设计即完成从操作型系统的数据表中抽取、转换、清洗以及将细节数据聚合为不同综合层次的数据,主要包括:

- 扫描模块的设计。扫描模块用于对现有业务系统进行有效的扫描,以便获取需要追加的数据集合。
- 定义数据转换和清洗规则。数据仓库需要从多个不同的数据源中抽取数据,不同数据源的数据具有不一致性,良好的数据转换和清洗规则是数据质量的重要保证。
- 数据抽取模块的设计。将需要追加的数据通过格式转换、清洗转换为数据仓库的细节表。
- 综合模块的设计。将细节数据聚合成各个综合层次的数据。

数据装载接口设计的重点是 ETL 设计,下面将详细加以介绍。

目前,数据仓库接口的实现方法有两种:一是程序员自行编写设计处理集成的接口程序,即通过代码实现 ETL 过程;二是购买 ETL 软件。

如果将数据仓库数据模型设计比喻为一座大厦的设计蓝图,数据是砖瓦,则 ETL 就是建设大厦的过程。在整个数据仓库的实施过程中,用户需求分析和模型设计是最难的,而 ETL 的设计和实施则是工作量最大的,约占整个项目的 60%～80%,这是国内外众多实践中形成的普遍共识。

1. 主要任务

ETL 的主要任务包括如下几方面的内容。

1) 数据抽取

这一部分需要在调研阶段做大量的工作,首先需要了解数据来自哪几个业务系统,各业务系统的数据库服务器运行哪种 DBMS,是否存在手工数据,手工数据量多大,是否存在非结构化的数据等,当收集完这些信息后才可以进行数据抽取的设计。

- 与存放 DW 的数据库系统相同的数据源的处理方法

对于这一类数据源,设计上比较容易。一般情况下,DBMS 都提供数据库链接功能,在 DW 服务器和原业务系统之间建立直接的链接关系则可以通过 SQL 语句直接访问。

- 与 DW 数据库系统不同的数据源的处理方法

对于这一类数据源,一般情况下也可以通过 ODBC 的方式建立数据库链接,如 SQL Server 和 Oracle 之间。如果不能建立数据库链接,可以通过两种方式完成:一是利用工具将源数据导出到.txt 或者.xls 文件,然后再将这些源系统文件导入 ODS;二是通过程序接口完成。

• 文件类型数据源(.txt、.xls)

可以培训业务人员利用数据库工具将这类数据导入到指定的数据库,然后从指定的数据库中抽取。或者借助工具实现,如 SQL Server 的 SSIS 服务的平面数据源和平面目标等组件导入 ODS。

• 增量更新的问题

对于数据量大的系统,必须考虑增量抽取情况下,业务系统记录业务发生的时间作为增量标志,即每次抽取前首先判断 ODS 中记录最大的时间,然后根据这一时间去业务系统抽取大于这一时间的所有记录。一般情况下,业务系统没有或者部分有时间戳。

2) 数据清洗

数据清洗(Data Cleansing,Data Cleaning,Data Scrubbing)是一个减少错误和不一致性,解决对象识别的过程。一般情况下,数据仓库分为 ODS 和 DW 两部分。通常是从业务系统到 ODS 需要进行清洗,即过滤脏数据和不完整的数据;再从 ODS 到 DW 需要进行转换,即进行一些业务规则的计算和聚合。数据清洗的任务是过滤那些不符合要求的数据,将过滤的结果交给业务主管部门,确认是否直接过滤还是由业务部门修正后再进行抽取。数据清洗的目的是保证数据仓库的数据质量。

数据质量定义为数据的一致性(Consistency)、正确性(Correctness)、完整性(Completeness)和最小性(Minimality)四个指标在系统中得到满足的程度。其中:

• 一致性,是指数据的值和描述在全局即数据仓库中均表示相同的含义。
• 正确性,是指数据的值和描述一定是真实的和业务系统保持一致的。
• 完整性,是指确保每一条数据都必须是有意义的(不能为 NULL 值)。
• 最小性,是指数据的值和描述有且仅有一个含义。

根据处理的是单数据源还是多数据源以及是模式层的还是实例层的,可将数据质量问题划分为四类:单数据源模式层问题(如缺少完整性约束、糟糕的模式设计等)、单数据源实例层问题(如数据输入错误)、多数据源模式层问题(如异构数据模型和模式设计等)、多数据源实例层问题(如冗余、冲突和不一致的数据等)。单数据源中出现的问题在多数据源中也可能出现,并且这种现象是普遍的;模式层的问题也会体现在实例层。模式层的问题可以通过改进模式设计、模式转化和模式集成加以解决;但实例层的问题在模式层上是不可见的,所以数据清洗主要针对实例层的数据质量问题。

实际上,数据清洗是利用诸如数理统计、数据挖掘或预定义的数据清洗规则等将脏数据转化为满足要求的数据。

按照数据清洗的实现方式和范围,可将数据清洗划分为四种类型,即:

• 手工实现方式。用人工检测所有的错误并改正,只适用于小数据量的数据源。
• 编写专门的应用程序。通过编写程序检测/改正错误。但通常数据清洗是一个反复的过程,将导致清理程序复杂、系统工作量大。
• 某类特定应用领域的,如运用概率统计方法查找数值异常的记录。
• 与特定应用领域无关的,主要集中于重复记录的检测/删除。

数据清洗是 ETL 的一个重要环节,其主要任务是检测并删除/改正将装载到数据仓库的脏数据。由于数据仓库的多种异构数据源和海量数据,数据清洗应是与领域无关的,而且数据清洗不是 ETL 中一个独立的步骤,需要与数据抽取、转换/集成和装载协同配合,并循

环反复进行。如果数据源是一个功能较强的 DBMS(如图 2.18 所示的数据源 1 和数据源 2),则可以在数据抽取时使用 SQL 完成一部分的数据清洗工作,但是有一些数据源不提供这种功能(如图 2.18 所示的数据源 3),只能直接将数据从数据源抽取出来,然后在数据转换时进行清洗。数据仓库的数据清洗主要还是在数据转换时进行的,使用 DBMS 的转换清洗功能完成大部分的工作,这样数据清洗就充分利用了 DBMS 提供的功能。

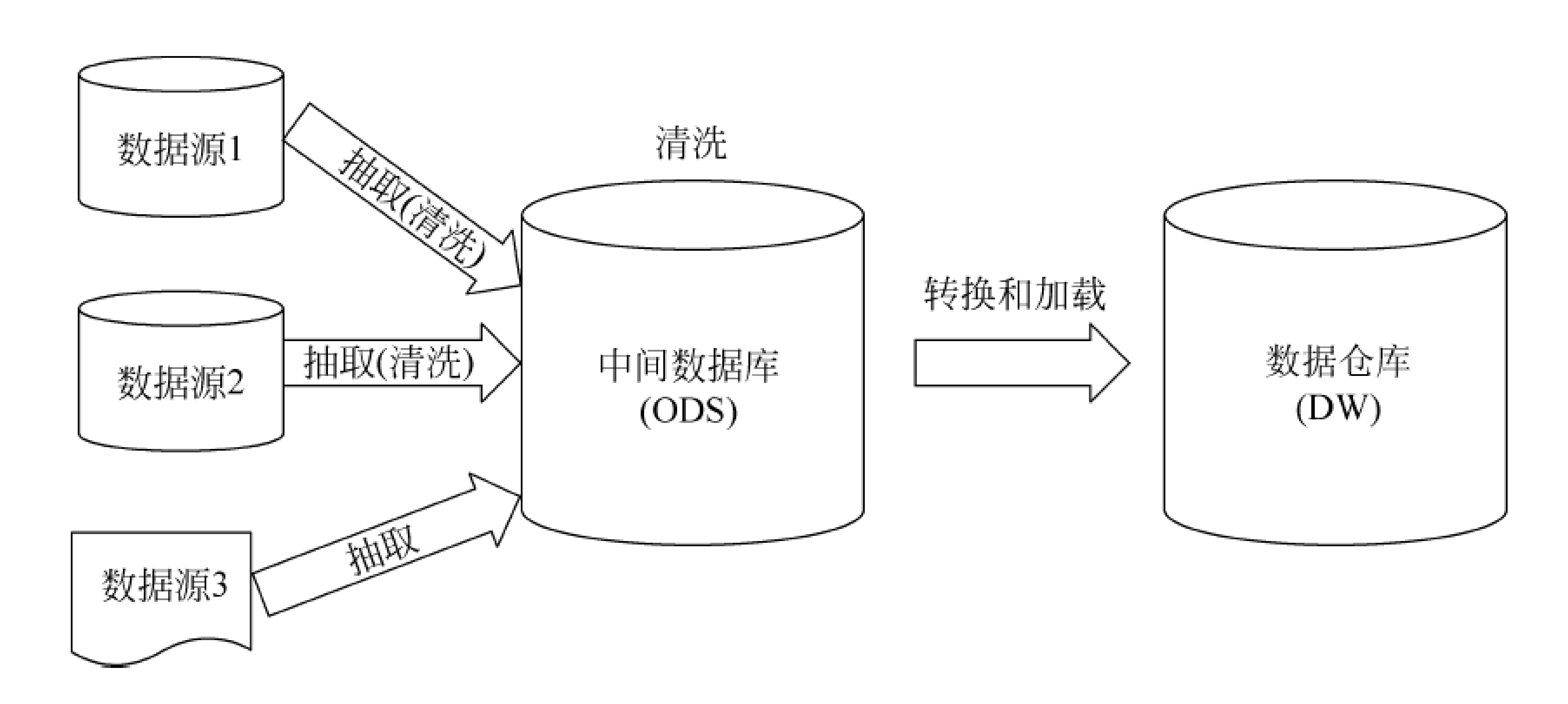


图 2.18　数据清洗过程

数据清洗是一个反复的过程,不可能在几天内完成,只有不断地发现问题、解决问题。对于是否过滤,是否修正往往要求用户确认。对于过滤掉的数据,写入 Excel 文件或者将过滤数据写入数据表,在初始阶段可以每天向业务部门发送过滤数据的邮件,促使他们尽快修正错误,同时也可以作为将来验证数据的依据。

3) 数据转换

数据仓库抽取数据的过程是从业务数据库开始的,中间会经过数据的转换过程,变成数据仓库需要的数据。在进行数据转换之前,必须首先进行数据映射(Source Data Mapping),明确定义数据仓库的每个表、每个字段来自源系统或接口单元中的哪张表、哪个字段。这种映射关系可能很简单,例如表是完全一对一的关系,直接复制到数据仓库即可;也可能很复杂,数据仓库的某张表可能来源于源系统中的多个表,这些源表通过一定的关系被关联起来,然后对表中的一些字段进行转换后,变成数据仓库中目标表的对应字段。数据转换的任务是对不一致数据的转换、数据粒度的转换以及一些业务规则的计算。

- 不一致数据的转换:这是一个整合的过程,将不同业务系统的相同类型的数据统一。例如同一个代理商在结算系统的编码是 XX0001,而在客户关系管理(Customer Relationship Management,CRM)系统的编码是 YY0001,这样在抽取后统一转换成同一个编码。
- 数据粒度的转换:业务系统一般存储细节数据,而数据仓库的数据是用来分析的,不需要细节数据。一般情况下,会将业务系统的数据按照数据仓库的粒度进行聚合。
- 业务规则的计算:不同的企业拥有不同的业务规则和数据指标,这些指标有时不是简单的运算能够完成的,需要在 ETL 中将这些指标进行计算后存储在数据仓库中,供分析使用。

4）数据装载

数据装载是将从数据源抽取、转换、清洗后的数据装载到数据仓库中。数据装载策略需考虑装载周期以及数据追加策略两方面的内容。根据业务数据的实际情况，装载周期应综合考虑业务分析需求和系统装载代价，对不同业务系统的数据采用不同的数据装载周期，但必须保持同一时间业务数据的完整性。数据追加策略根据数据的抽取策略以及业务规则确定，一般分为直接追加、全部覆盖和更新追加三种类型。

- 直接追加是指每次装载时直接将数据追加到目标表。
- 全部覆盖是指如果抽取数据本身已经包括数据的当前和所有历史状况，可对目标表采用全部覆盖的方式。
- 更新追加是指对于需要连续记录业务的状态变化，并用当前最新状态与历史状态进行对比的情况，可以采用更新追加的方式。

2. 设计原则

由于数据源的多样性，数据传输条件不确定性以及用户对最终统计数据的选择性等因素，使得 ETL 设计不仅要考虑业务数据处理的要求，还应考虑数据传输过程中如何解决上述问题。ETL 设计一定是针对具体应用，不同的业务和不同的分析模型的要求，所以 ETL 整体架构的灵活性和扩展性非常重要。

ETL 设计应遵循的原则如下：

（1）ETL 设计前需要根据业务特点确定分析主题和分析模型的结构，区分维度数据和事实数据，建立相应的数据仓库模型。在设计过程中需要考虑是否需要预留字段、增加属性等。

（2）数据粒度在同一立方体（Cube）中必须统一。事实表的粒度就是维表与事实表相关联的最小级别，尽可能采用粗的粒度以有效减少数据量，但是不同的分析可能存在差异，例如话务量趋势预测的主题中可以以小时为粒度，但是在一些实时性要求较高的忙时话务量分析主题中需要采用 15 分钟甚至 5 分钟的粒度。

（3）数据周期的确定，在 ETL 设计时需要事先确定抽取的时间，这可能需根据用户对实时性的要求作为调整的依据。

（4）尽量采用增量的抽取方式以减小每次抽取的数据量。

（5）数据流和工作流的概念。在 ETL 中需要考虑数据在每个步骤中的状态和转换行为，数据的清洗、转换和加载过程是由很多步骤完成的，每一步骤一定是数据的一个原子业务操作。步骤可以根据需要进行调整，在数据流可能会出现分支的情况，即在不同条件下采用不同的处理步骤。

（6）流程的异常处理。在流程中需要考虑各种可能的异常，如网络闪断、数据的延迟等。因此，需要在流程中加入一些补救措施予以纠正，保证导入数据仓库数据的绝对正确（但是流程的设计尽量依照简单、高效的原则）。

（7）ETL 的调整、运行管理以及监控。ETL 程序的运行应该有相关的管理和监控工具。一方面用于 ETL 的设置和调整；另一方面也是便于 ETL 出现异常时能够及时通过人工方式干预，保证 ETL 正常运行。

（8）针对业务需求进行 ETL 的配置和设置，方便专业维护人员和开发人员对抽取任务进行调整和灵活配置。

(9) Cube 的管理。除了对数据仓库的管理和数据的处理之外,ETL 的一项很重要的工作是对 Cube 的管理。根据 Cube 特性考虑对维度和 Cube 的更新,以及对 Cube 分区的新建、处理和合并等一系列操作。

(10) 数据仓库有一个初始化的过程,即将以前的业务数据进行整理和加载,但是数据量非常巨大,需要花费较长的时间,而且抽取的策略与平时的不同。

(11) 程序具有自修复功能。在任何一个步骤出现异常,ETL 程序都能够回到抽取前的状态,而不需要人工干预,更不能影响已抽取的数据。

3. 主要步骤

下面介绍 ETL 设计的主要步骤。

1) 设计数据准备区的数据结构

数据准备区是 ETL 专门用于对数据进行抽取、清洗和转换等处理的临时数据库。需要根据实际的需求设计数据准备区的库表结构。

2) 定义数据抽取规则

数据抽取首先需要定义数据抽取规则,记录在"数据抽取规则表"中,然后再设计数据抽取流程。

3) 定义数据清洗规则

数据清洗是为了处理数据源中存在的错误、不一致或无用的数据,即"脏数据"。在清洗"脏数据"之前,必须清楚存在哪些"脏数据",记录在"脏数据登记表"中。对发现的"脏数据"逐一确定清洗转换规则,记录在"清洗转换规则表"中。

4) 定义数据转换规则

数据转换是将抽取的数据进行过滤、合并、解码和翻译等,为数据仓库创建有效数据的过程。转换的过程需要理解业务侧重点、信息需求和目前可用的源数据。通常数据抽取完毕后,应根据企业业务的具体需求,设计和定义一系列的数据转换规则,转换规则主要包括:

- 字段级。定义数据类型转换、增加"上下文"数据,如时间戳。
- 多数据源整合。字段映射(Mapping)、代码变换(Transposing)将不同数据源中的数据值规范成数据仓库的数据值。例如,将源系统非英文编码转换成数据仓库英文编码,将源系统信息编码转换成数据仓库信息编码等。合并(Merging)将两个或更多源系统记录合并成一个输出或目标记录。派生(Derivation)根据源数据,利用公式产生数据仓库需要的数据。例如用身份证号码计算生日、性别和年龄等。
- 聚合(Aggregation)和汇总(Summarization)。

5) 设计 ETL 流程

ETL 流程设计是定义 ETL 流程的步骤,并确定每一步骤需要完成的工作,以流程图的形式加以描述。

4. 工具

ETL 实现方式有两种,即购买 ETL 工具和手工编程。表 2.2 给出了这两种方式的比较,为企业最终根据实际情况选择合适的方式提供参考,ETL 工具早期的数据迁移大多是由开发人员手工编程实现的。

表 2.2　购买 ETL 工具和手工编码实现 ETL 的比较

性　　能	ETL 工具	手 工 编 码
灵活性	比较灵活	非常灵活
难易程度	相对容易	要求一定技术水平
管理和维护	容易	较难
性能和效率	较高	取决代码的质量
开发周期	较短	较长
工作量	中等	较重
价格	较高	相对较低

选择 ETL 工具时应考虑可能影响 ETL 功能的主要因素，主要包括平台的支持；数据转换功能；管理和调度的功能；集成和开发性以及对元数据的管理等。目前，市场上主流的 ETL 工具分为两大类：一类是 ETL 厂商提供的产品，一般都具有较完善的体系结构和功能。典型产品包括 Ascential DataStageXE 和 Informatica；另一类是数据库厂商提供的整体数据仓库解决方案和产品，这类产品在提供数据仓库存储、设计和展现工具的同时提供相应的 ETL 工具，它们对相关产品具有很好的支持并能发挥出最大效率，但结构封闭，对其他厂商产品的支持有限，如 Oracle Warehouse Builder 和 IBM Warehouse Manager 等。

评价 ETL 工具性能的主要因素包括：

- 可靠性——系统具有高的容错性和故障恢复能力，具备完善的备份、恢复等机制保证系统的稳定性和可用性。
- 可衡量性——系统的性能指标可以根据系统的软硬件配置进行调整，系统扩展性强。
- 性能——系统具备并行计算、负载调度等保障系统性能的良好机制和能力。
- 代码生成能力——是指将图形化操作生成的 ETL 过程转换为标准 SQL 或系统专用存储过程的能力。
- 开发——系统提供的开发接口的通用性、灵活性、功能性决定了系统开发的工作量和难度。
- 元数据管理——系统可以维护 ETL 中的技术元数据和业务元数据。
- 管理——ETL 的管理包括系统的安装、配置和维护。

四种常用 ETL 工具的综合比较如表 2.3 所示。

表 2.3　四种常用 ETL 工具的综合比较

	可靠性	可衡量性	性能	代码生成能力	开发	元数据管理	管理
Power Mart	B	B	A	A	A	B	A
DataStage	B	B	A	A	A	B	A
OWB	B	B	C	B	B	C	B
DB2 WM	B	A	C	B	B	C	B

（注：A—优秀；B—较好；C——一般）

2.2 数据仓库实现

1. 数据仓库的实现方法

数据仓库的实现方法主要包括自顶向下、自底向上、平行开发、有反馈的自顶向下、有反馈的自底向上等，这里主要介绍当前业界流行的自顶向下、自底向上和平行开发三种方法。

1) 自顶向下

这是一种由整体到局部，逐步细化的实现方法。首先对分散在各业务数据库的数据特征进行分析，在此基础上实施数据仓库的总体设计和规划，准备元数据。随后，进行外部数据源的数据抽取、转换和加载等一系列处理，并将处理后的数据导入数据仓库，元数据也同时导入，从而建立一个完整的数据仓库，并针对各个主题建立数据集市，以满足分析决策的需求，如图 2.19 所示，其中数据集市是数据仓库的真子集。数据仓库的实现过程直观、清晰、易于理解，只要对外部数据源和所支持的决策有较深入的理解，保证各数据集市都是数据仓库的真子集，则可以完全消除信息之间的"蛛网"现象。其不足之处是要求设计者对业务有深入的理解，系统设计规模偏大，实施周期过长，见效缓慢，尤其在项目实施初期见效不明显。

2) 自底向上

一般企业在构建数据仓库时，往往准备的数据规模偏小，决策目标不明确，并且希望数据仓库能较快地发挥作用，产生效益。为了满足上述要求，并克服自顶向下的不足，自底向上应运而生。

与自顶向下相反，自底向上的设计思路是先具体，后综合。首先，将企业内部各部门的需求视为分解后的决策子目标，并针对这些子目标建立各自的数据集市，从而获得最快的回报。在此基础上，对系统不断扩充，逐步形成完善的数据仓库，以实现对企业决策的支持。由于数据集市结构简单，数据的综合度较低，因此不需要创建数据仓库所必需的元数据部件。自底向上的实现方法如图 2.20 所示。其优点是投资小、见效快。由于部门级的数据结构简单，决策需求明确，因此易于实现。但是由于数据集市缺少元数据，最终构建数据仓库的过程具有相当的难度，并有可能影响数据仓库整体结构的合理性以及系统的运行效率。

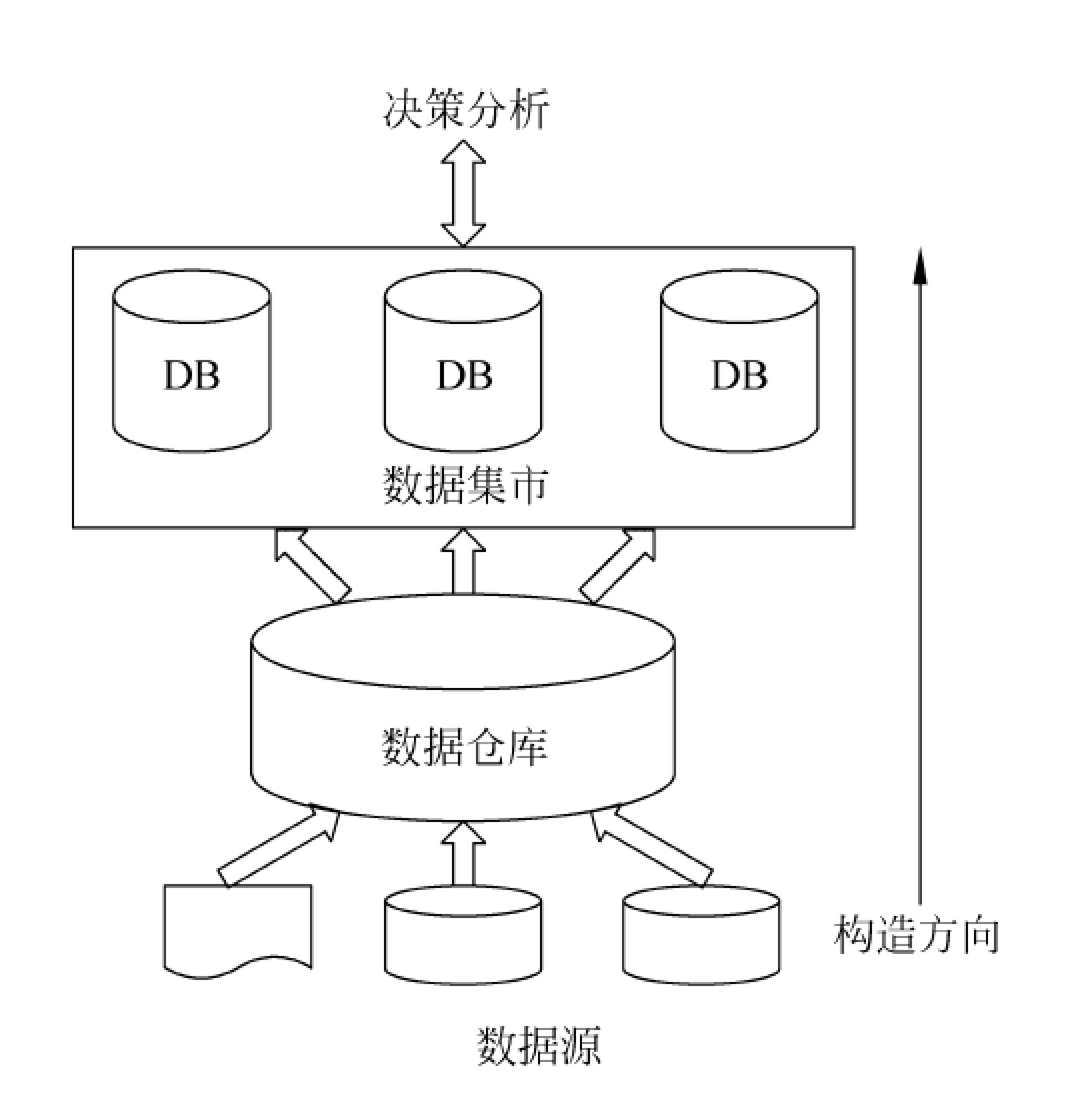


图 2.19 自顶向下的数据仓库实现方法

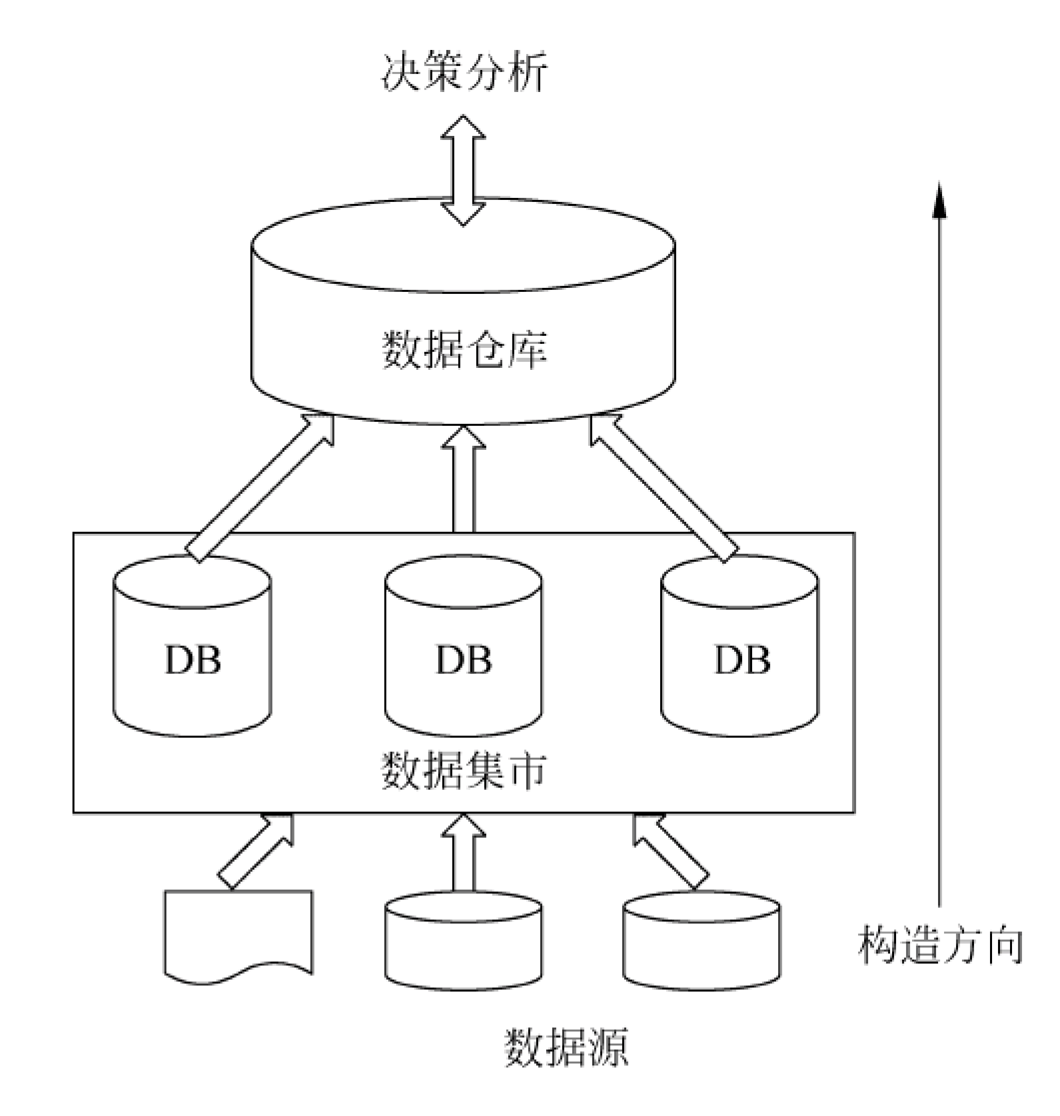


图 2.20 自底向上的数据仓库实现方法

3）平行开发

平行开发是指在同一个模型的指导下，建立数据仓库的同时，建立起若干数据集市，如图 2.21 所示。这种方法是在自顶向下的基础上，同时吸收了自底向上的优点，因此可以认为是两种方法的有机结合。

在平行开发中，数据仓库和数据集市遵循统一的数据模型的指导，同时建立、避免了建立相互独立的数据集市的盲目性，有效减少数据的不一致和冗余。其核心为两部分：一是统一的“全局元数据中心库”(GMR)，记录数据仓库的主题域、通用维、业务规则和其他各种元数据；其二是“动态数据存储区”(DDS)，存储从数据源中抽取的数据，并为进一步处理做好准备。GMR 和 DDS 不是一成不变的，它们都随着数据源以及决策需求的变化而改变。

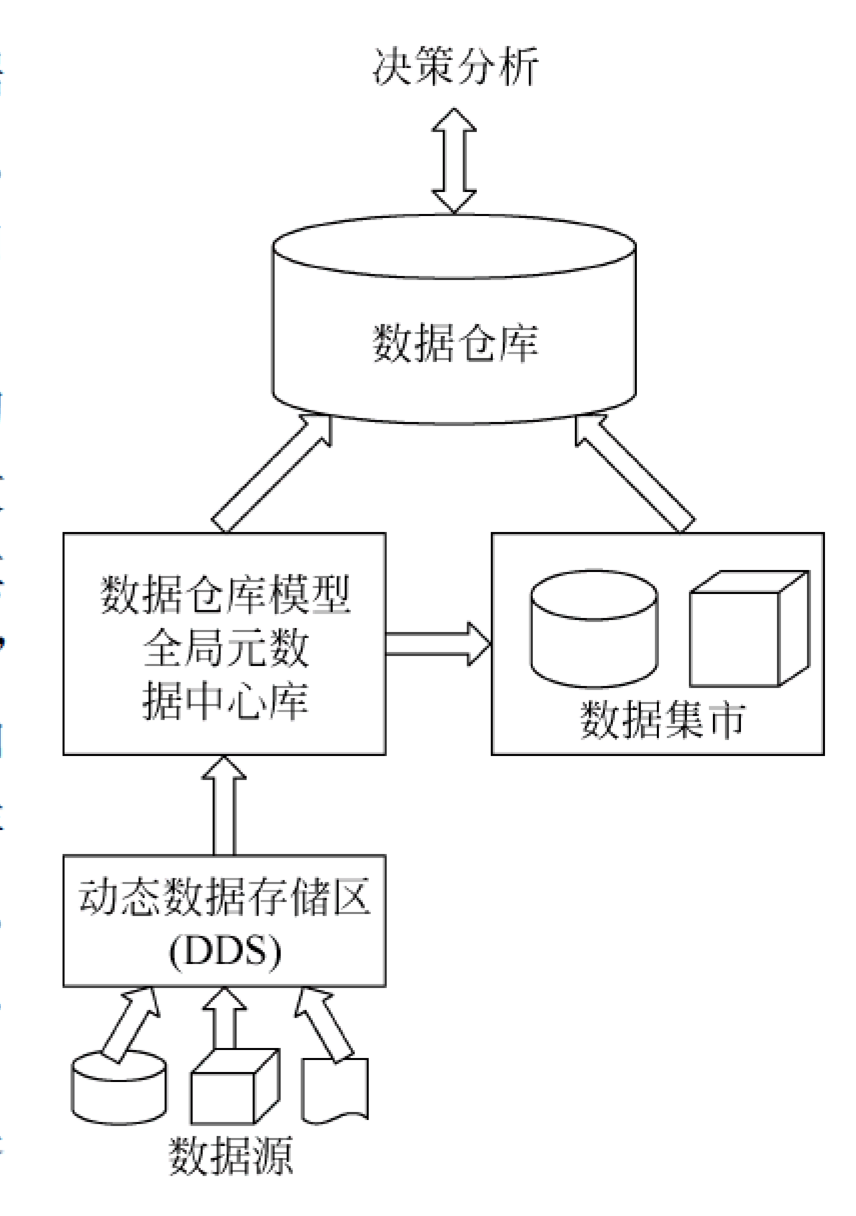


图 2.21　平行开发的数据仓库实现方法

第一种方法类似于软件工程中“自顶向下”，投资大、周期长。第二种方法恰恰相反，类似于软件工程中“自底向上”的方法，投资少、周期短且易见成效，但由于设计之初是以特定的部门级主题为框架，向其他的主题和部门扩充往往比较困难。实际中，大多采用第二种方法。

“自底向上”建立数据仓库，并不意味着不需要在设计阶段的长远规划。采用逐步积累的方式建立数据仓库，其最大的问题是已有的框架无法把新的业务集成进来。因此，在设计阶段就必须充分考虑到这一点。例如，部门级的主题是否有助于形成企业级的主题，数据加载模块是否能够重用等等。数据仓库的长远规划，并不仅仅是技术部门的事情，应当把构建数据仓库作为企业发展战略的一个组成部分。在设计阶段需要不同部门的沟通和协调，技术框架和系统设计必须从整个企业的角度加以考虑，即使刚开始实施时是面向某个部门的。从这一点来看，建立一个企业级的数据仓库，主要的障碍不是技术，而是不同部门之间的组织和协调问题。

2. 数据仓库的开发方法

目前，开发方法有多种，如生命周期方法、快速原型法和螺旋法等。虽然通常人们在开发数据库系统时多数采用生命周期法或原型法，但由于数据仓库固有的特点，因此传统的生命周期法和原型法并不适用于数据仓库的开发。

螺旋式的开发方法采用“分而治之”的思想，将一个庞大的任务划分成多个阶段。在每个阶段，项目按照问题定义、系统分析、系统设计、开发、实现、维护和系统评估来进行。一个阶段完成后，再开始新的阶段，而每一阶段都是以前一阶段的结果为参考点，再增加新的需求项目，直到所有的需求都满足为止。

螺旋式开发方法具有以下特点：

- 每一个区域均含有一系列适应待开发项目特点的工作任务。
- 适用于需求不断增长的系统开发。

与传统的开发方法不同，螺旋式开发不是当软件交付时就结束了，它能够适用于系

统开发和使用的整个生命周期,会一直运转到软件退役。有时这一过程处于休眠状态,但任何时候出现了改变,过程都会从合适的入口点开始继续运转。

一般地,数据仓库的原始需求并不明确,且不断变化与增加,开发者最初并不能确切了解用户明确而详细的需求,用户所能提供的无非是需求的大方向以及部分需求,不能较准确地预见到以后的需求。因此,数据仓库比较适合采用螺旋式开发方法,但是这又不同于一般意义上的螺旋开发法,数据仓库是在原有数据库基础上构建的,即从已经存在于操作型数据库环境中的数据出发构建数据仓库,即是“数据驱动”的,因此,恰当的称谓是数据驱动的螺旋式开发法,该方法是将数据驱动的思想与螺旋式开发方法结合起来的一种开发方法。基于“数据驱动”的主要特点是:

- 利用以前已经建立的数据库建设数据仓库。尽量利用已有的数据和代码,而不是从头开始,这是数据驱动思想的出发点。
- 基于数据驱动的方法不再是面向应用,而是面向主题。数据仓库的开发是从已有的数据库系统出发,按照分析领域的要求对数据及数据之间的关联重新考察,以组织数据仓库中的主题。

基于数据驱动的螺旋式开发方法是一个不断向外扩展的迭代过程,其起点较低,每次迭代,螺旋线增加一周,数据仓库的开发又前进一个层次,系统又生成一个新版本,而软件开发的时间和成本又有新的投入。在螺旋式前进的过程中,最后总能得到一个用户满意的软件版本。所以基于数据驱动的螺旋式开发方法,非常适于数据仓库面向主题、基于数据驱动的开发特点。该方法也是目前数据仓库常用的开发方法。

数据仓库的建立是一个数据驱动、技术支撑并满足应用需求不断增长的开发过程。数据仓库的开发像生物一样具有其特有的、完整的生命周期。数据仓库的开发周期可以分为数据仓库的规划分析阶段、数据仓库的设计实施阶段以及数据仓库的使用维护三个阶段。这三个阶段是一个不断循环、完善和提高的过程。一般情况下,数据仓库不可能在一次循环过程中完成,而是经过多次循环开发,每次循环都会为系统增加新的功能,使数据仓库的应用得到新的提升。

3. 数据仓库的维护

数据仓库建成后就进入运行维护。一方面,用户使用数据仓库中的数据进行决策或者分析,即在数据仓库中建立 DSS 应用,同时,用户将使用情况和新的需求反馈给开发人员以进一步完善系统,并管理数据仓库的一些日常活动,如刷新数据仓库的当前详细数据,将过时的数据转换为历史数据,清除不再使用的数据,并调整数据粒度级别等。

数据仓库维护的首要任务是数据备份与恢复。数据仓库的数据是多年积累的结果,可能包括 10 年甚至 20 年的历史数据。这些数据代表了企业浓缩而丰富的历史。构建一个成功的数据仓库投入的资源无疑是巨大的,所以数据仓库的数据一旦丢失,将给企业造成重大损失。为避免这种灾难的发生,需要对数据不断备份。

1)备份

备份是数据安全的保证。实现备份需要考虑以下一些因素:

- 确定哪些部分需要备份。将当前数据和历史数据分离,当前数据源随操作型系统的输入而增长,历史数据是过去的内容,应经常性地备份当前数据,历史数据没有必要频繁备份。

- 数据仓库的容量是一个大问题。完全备份数据仓库需要很长时间，除了完全备份，还可以考虑日志备份和差异备份。
- 周期性地归档数据。在数据仓库中周期性地将非常陈旧的数据归档，以减少备份和恢复的时间，提高检索的性能。
- 备份时间。OLTP 系统备份一般在晚上执行。但对于数据仓库晚上的时间用于每日的增量装载，可以考虑备份和每日装载同步进行。如果可用，将增量装载文件作为备份的一部分存储。
- 备份介质的选择。备份介质的选择很重要，这取决于数据仓库的容量。

2）恢复

当数据仓库发生崩溃时，可以利用备份文件恢复系统。一般地，恢复过程需要注意以下问题：

- 明确的恢复计划。将不同的灾难情况列表，指出每种情况下如何恢复。
- 考虑公司的条件，建立恢复步骤，估计恢复的期望停机时间，正确、迅速地通知用户。
- 如果必须在源系统完成恢复过程，应保证源系统可用。

构建数据仓库是一个长期工作，与其他系统一样需要在运行过程中不断调整和完善。其次是性能的优化。数据仓库涉及海量数据的查询，数据的大量写入/读出，不仅对系统要求很高，而且与 OLTP 的要求极为不同，因此在数据仓库设计、开发、实施和维护的过程中，数据仓库系统的性能都是一个不容忽视的问题。尤其是在运行期间，应密切关注应用对系统资源的消耗情况，针对应用的特点及时进行系统的调整，包括调整数据库参数、数据分片设置、创建特殊索引乃至提高系统配置等。

此外，还可以考虑模型的调整。应用与需求是相互促进、不断发展的，随着系统建成并运行，用户在对系统了解不断加深的过程中，也会对系统提出更新、更高的要求。如何在最小投入的前提下满足用户的需求，也是一个值得注意和潜心研究的问题。应尽可能挖掘现有系统的潜力，其次考虑对主题的增加或可在现有系统上增加少量指标，对系统进行适当调整，最后才考虑系统重构，尽可能减小系统建设的投入。

第 3 章　数据仓库实例

3.1　实例一

随着数据仓库技术的成熟，国外数据仓库技术已广泛应用于电信、金融和保险等行业。近年来，中国移动、中国电信和中国联通等各大电信运营商纷纷制定出经营信息服务系统技术规范和业务规范，开始建设企业级数据仓库系统，引领国内数据仓库应用的发展。

在“全业务”运营模式下如何发挥综合优势，实现灵活多变的市场营销策略，为客户提供更具有针对性的服务；同时最大限度地合理配置和优化自身资源，降低运营成本，以增强企业的核心竞争力，成为当前迫切需要解决的问题。其中，市场策略的应对速度是保持竞争力的重要手段，而市场应对策略的制定需要丰富、真实和及时的经营信息提供支持。因此，构建数据仓库势在必行。下面将详细介绍如何构建一个简单的面向电信领域的数据仓库。

3.1.1　选择主题

根据电信业务和运营的需求，其主要主题域如图 3.1 所示。

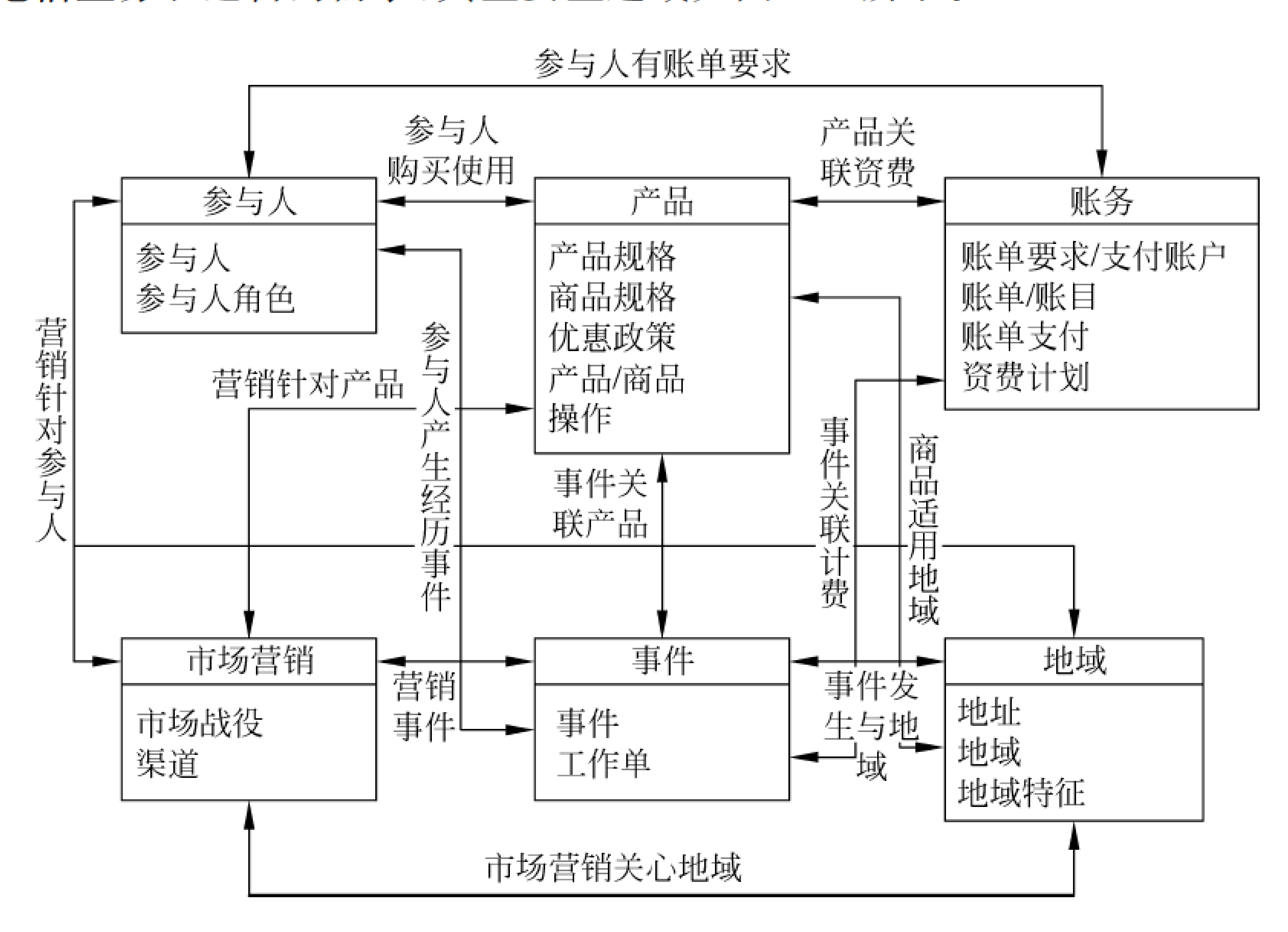


图 3.1　电信领域数据仓库的主要主题域

其中：

- 参与人是指与电信运营商的业务或管理活动存在联系的任何个人或组织机构，是一切电信活动的主体，包含电信企业的客户、联系人等所有消费电信产品的信息。参与人角色则描述了参与人在消费过程中扮演的角色。

- 产品主题域主要包括产品和商品两个概念。产品是指电信运营商利用自身的资源或者第三方资源，为客户提供具有市场价值的基本元素；商品是指电信运营商利用营销手段针对不同的营销渠道、客户细分、地域细分和销售目标等，对产品规格、资费计划进行必要的组合包装的产物。产品是电信企业提供的基本服务单元，如固定电话、小灵通、专线等；商品是指给产品制定了资费以后的服务。参与人直接购买使用商品。
- 账务主题域是指参与人使用商品过程中所产生的费用、账目等，这些账目的资费是根据商品的资费进行计算的。
- 市场营销主题域主要包括市场战役和渠道。市场战役主要描述企业在市场营销活动中的策略和效果；渠道是企业利用营销的管理手段管理企业客户的方法。
- 事件主题域给用户提供了一个客户生命周期的完整视图，记录了客户与电信运营商关系的不同阶段。
- 地域主题域是指在地理上能被确定的一块地方，是一个面的概念，可以是行政区域也可以是电信的管理区域。

通常地，根据需求选择用户最关心的主题域，同时还要兼顾业务系统的数据提供能力。

3.1.2 逻辑模型

本例中，逻辑模型的设计主要包括以下几个方面。

1. 选择数据源

构建面向电信领域的数据仓库，数据源主要是来自于各类业务以及营业、账务和计费等生产系统的数据，如客户资料、通话详单和出账数据等。

表 3.1～表 3.3 分别给出了移动业务的客户资料表、出账表和通话详单表的常用字段。

表 3.1 客户资料表的常用字段

客户标识
手机号码
客户类别
客户姓名
证件类型
客户证件号码
归属地
付费方式
入网日期
…

表 3.2 客户出账表的常用字段

客户标识
基本月租费
增值服务费
本地通话费
长途通话费
国内漫游通话费
国际漫游通话费
短信费
总费用
…

表 3.3 通话详单表的常用字段

呼叫类型
IMSI 号
主叫号码
被叫号码
通话开始时间
通话时长
通话位置
漫游类型
…

2. 确定数据粒度

数据仓库设计最重要的步骤之一是确定数据粒度。

1) 选择数据粒度

由于电信业务系统中的数据量很大,例如某省一天的 GSM 业务详单数据量就达 1.2GB 左右,客户账单的数据量每个月也有 2000 万条,同时对详单数据保留周期要求在线保存 3 个月,其他如客户资料、账单和欠费等详细数据的保留周期更长,因此数据仓库系统应采用多重粒度,使数据在当前细节级和各个汇总级都存在,以满足用户的不同查询需求。

对于客户资料,由于属于增长较为缓慢的信息(随着客户数量的增长,客户业务信息的变更表会增长),可以使用单一的数据粒度。如果客户数量很大,每个月生成一张客户资料表将浪费大量的空间,因为只有客户资料变化或者新增客户,才需要修改原先的客户资料表,采用增加一个"当前标志"字段,标识客户的当前信息,并选择"数据变更日期"作为时间字段,如图 3.2 所示。

对于客户出账信息,每月每个客户标识只在账务信息中对应一条记录,本身就带有一定的综合性,可以采用单一的数据粒度,如图 3.3 所示。

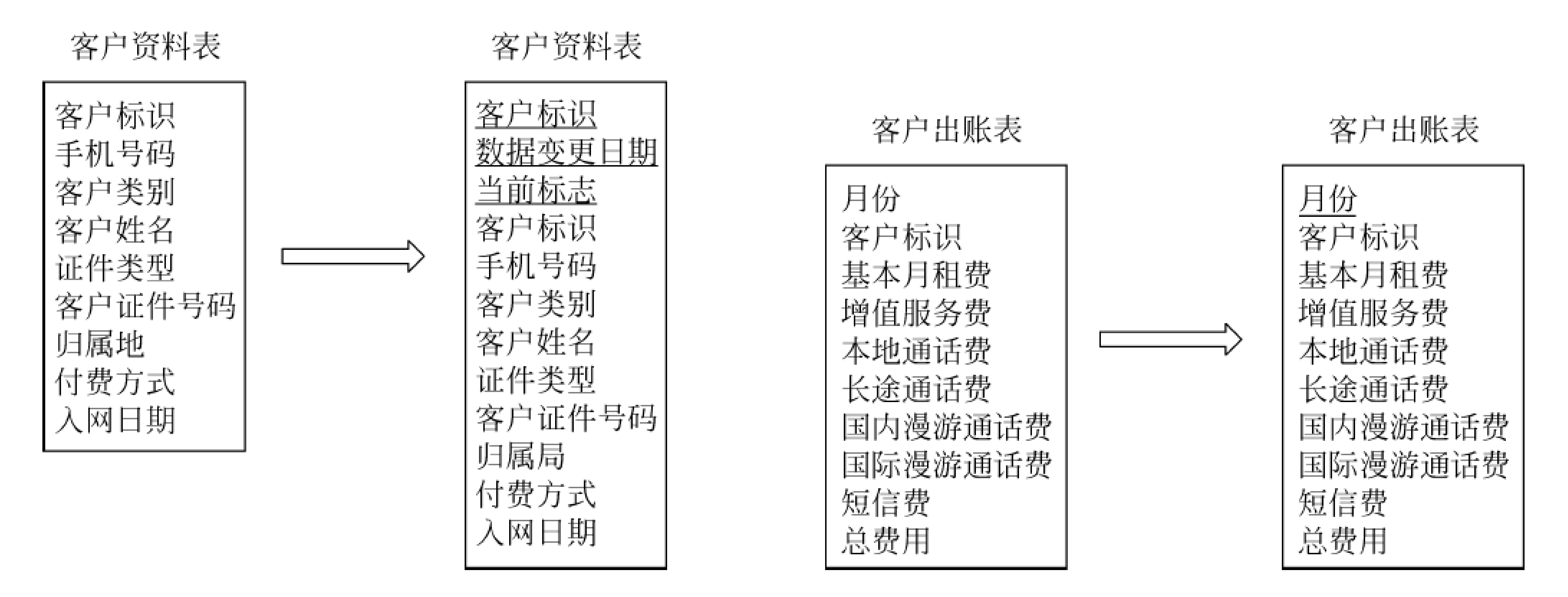


图 3.2 使用单一粒度的客户资料

图 3.3 使用单一粒度的客户出账数据

通话详单数据量最大,对于一个客户的一次通话(无论是主叫或被叫,因为一次通话实际上将生成主叫、被叫两条记录),通话详单中将出现一条通话记录,因此对于一个大型的电信公司,其通话详单数据量非常大,所以采用双重粒度。

对于近 3~4 个月的通话详单数据,保留在数据仓库中,并定期聚合成按月综合,然后将细节数据导出另行保存为新的细节数据腾出空间,如图 3.4 所示。

2) 选择粒度级别和汇总方式

对粒度级别的选择主要取决于用户的分析需求和各个表中数据量的大小。其中用户需求主要来自在需求分析阶段得到的各种用户报表和对分析需求的描述资料。针对不同数据层次选择粒度级别如下:

(1) 当前细节级

保存业务系统的最详细数据,不进行数据汇总。

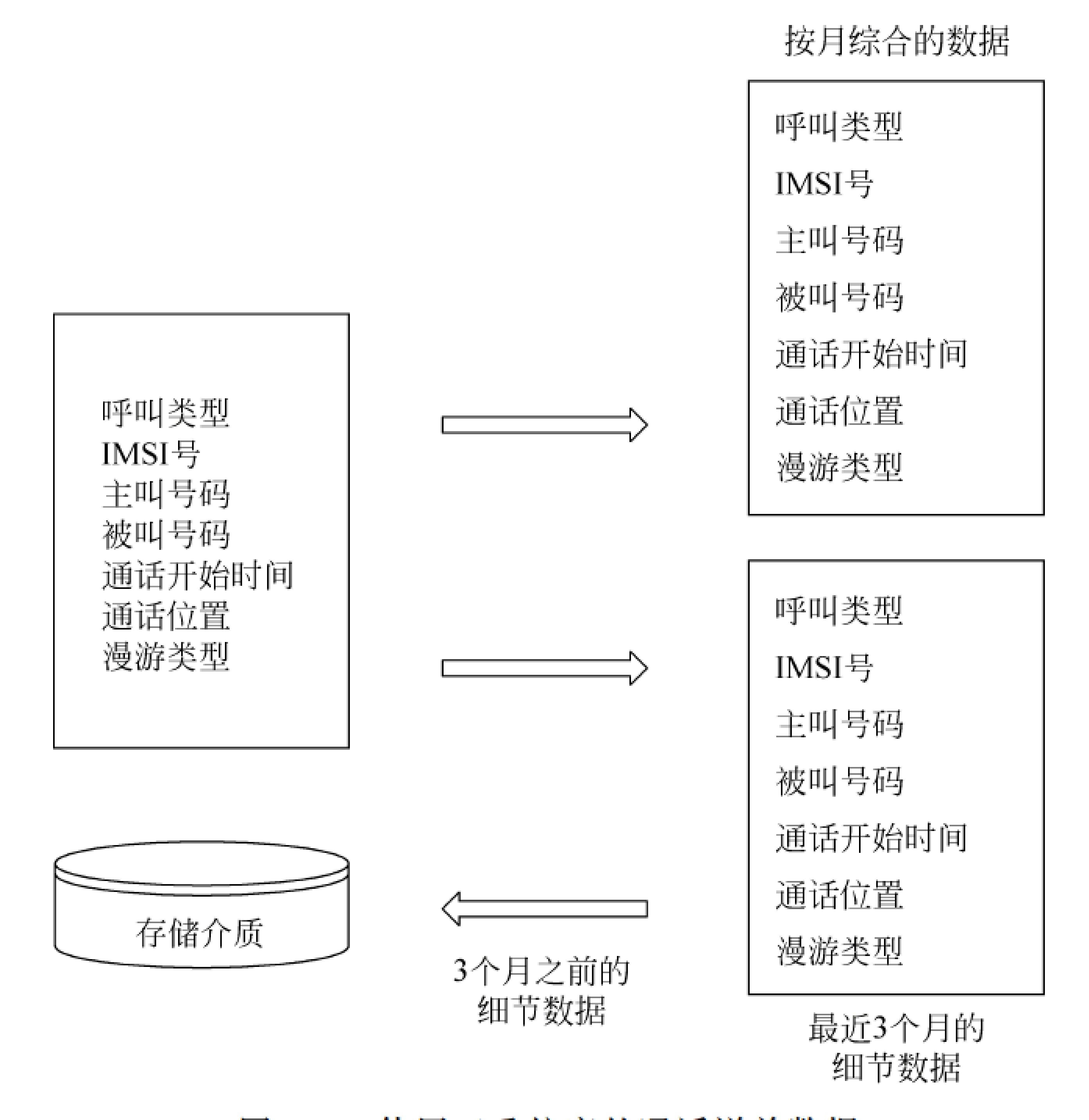


图 3.4　使用双重粒度的通话详单数据

(2) 轻度综合级

按照日和月份汇总数据，其中，话单数据是按照日和月份汇总一个用户的话务数据，同时保存各种话务类型维度，由于按照这种汇总方式汇总后的数据量仍然很大，同时有对用户的月通话时长进行分类的要求。因此，增加每日和每月每个用户的通话总次数和总时长的汇总表；账单数据是汇总一个用户一个月的费用，去掉细节表中的最小费项，改变账单数据模式，将用户标识和账期作为主键，把主要费项作为字段加以保存；欠费数据是按照日和月份对用户欠费进行一次快照，同时汇总每个用户的欠费数据；销账数据是按照日和月份汇总用户的销账数据。

(3) 高度综合级

高度综合数据一部分是对轻度综合数据按照时间汇总到年，同时由于需满足用户的分析需求，带有用户标识的汇总数据量还是很大，因此在高度综合级增加一个不包括用户标识的汇总数据以再次减少数据量，提高查询分析的效率。这部分数据模型通过对用户已有报表的指标分析和用户提出的分析需求确定。

3. 设计数据模型

一般地，ODS 存放数据仓库的当前细节级数据，采用满足第三范式的数据模型。因为 ODS 的一部分数据如用户数据、客户数据等都需要从接口中提取变化的数据对表进行更新，因此使用满足第三范式的数据模型，可减少数据冗余，便于数据更新。同时还减少了在向 ODS 抽取数据时进行关联操作的次数。另外，由于 ODS 是轻度综合级数据的数据源，因此为了保证轻度综合级数据抽取的效率和准确，可在 ODS 中适当引入冗余字段。

在客户主题域的轻度综合级中对每个客户的数据进行汇总，其数据量很大，例如某省 GSM 业务的客户数大约是 300 万，如果采用星型模式，由于很多维度中成员数量很少，同时

轻度综合级的数据又不断增加。如果完全采用星型模式将造成数据大量冗余,增加系统存储压力,因此,对轻度综合级数据采用星型和雪花型模式的综合,将用户查询中经常用到的维度直接放到事实表,其他维度和相应的维表关联,这样即可以提高一定的查询效率,同时又不会造成过多的数据冗余。用户查询时,使用最多的是业务类型、用户所属地域和套餐信息等,以及付费方式和用户账务类型等派生字段,因此在各个轻度综合的事实表中加入业务类型、用户所属地域、套餐以及付费方式和用户账务类型,相应的维表直接和事实表关联,其他维表需要和用户维表关联。轻度综合级的数据模式如图 3.5 所示。

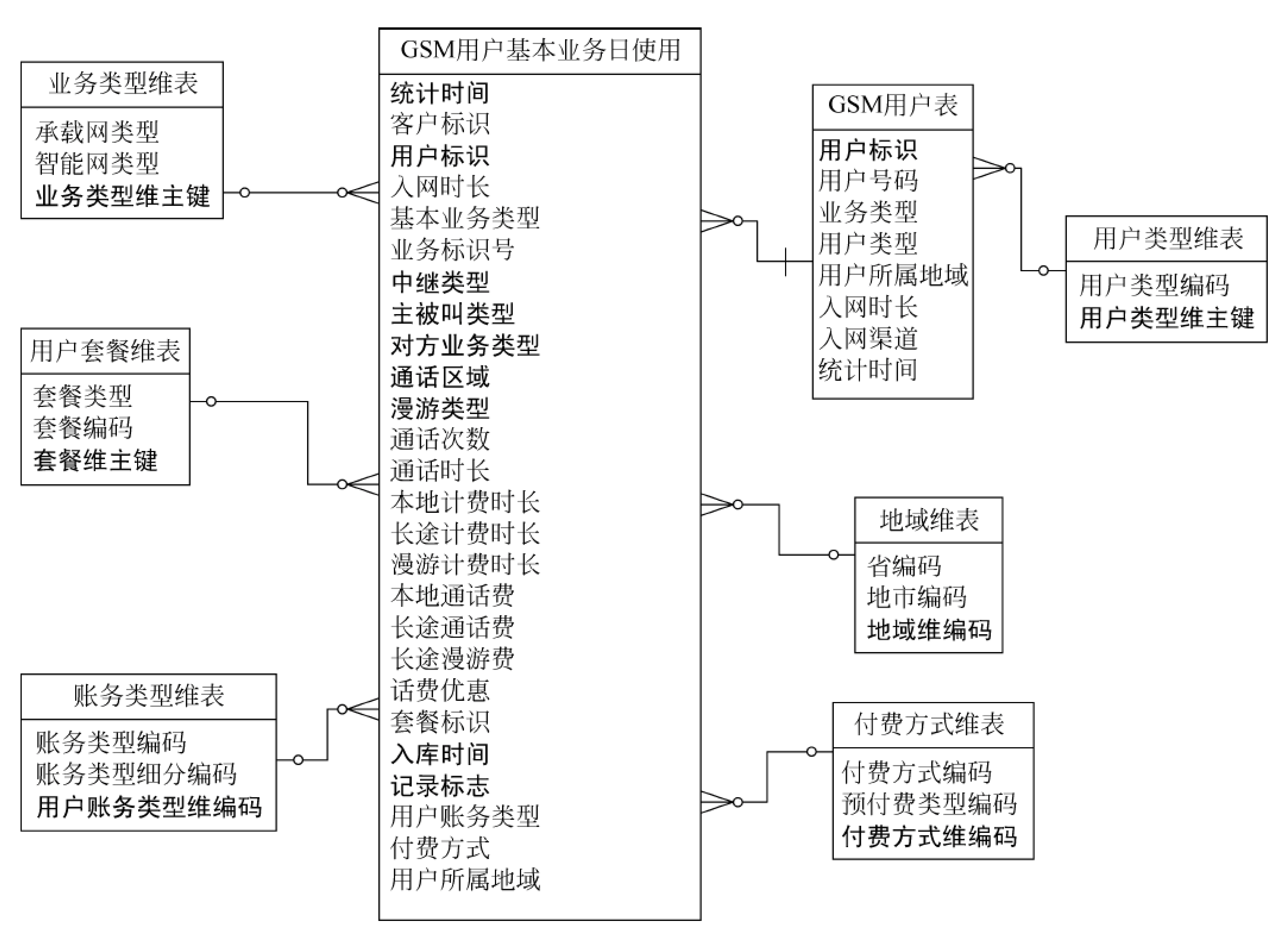


图 3.5　轻度综合级的数据模型——星型-雪花型

对于高度综合级的数据,一部分是对单一用户一年数据的汇总,因此也可采用星型和雪花型组合的模式。同时,在高度综合级还有一部分是不包括用户标识的汇总数据,数据量相对较少,因此可采用星型模式以提高数据查询效率。

设计维表时,应遵循以下的设计原则,即:

(1) 在维表中保存该维度的全部层次信息,同时引入代码表,对同一含义的字段进行统一编码,保证不同维表中相同含义的字段具有统一编码。维表和代码表的关系如图 3.6 所示。

(2) 维表只对最低层次编码,如果两个事实表对同一维度汇总到不同的层次,则对于该维度,不同的事实表对应不同的维表。例如,对于

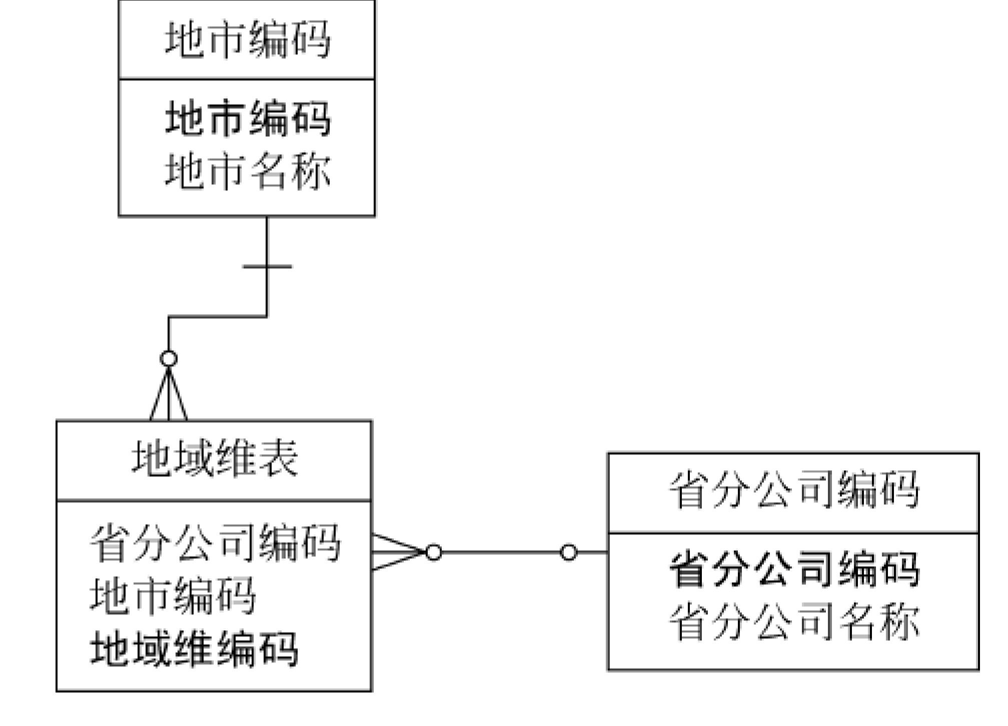


图 3.6　维表和代码表的关系

通话区域维，其层次分别为“省、地市、交换区、区县和基站”，通话情况的事实表有两个，其中一个在通话区域维的粒度是基站，而另一个在通话区域维的粒度是地市，则这两个事实表分别对应两个不同的通话区域维表，如图 3.7 所示。

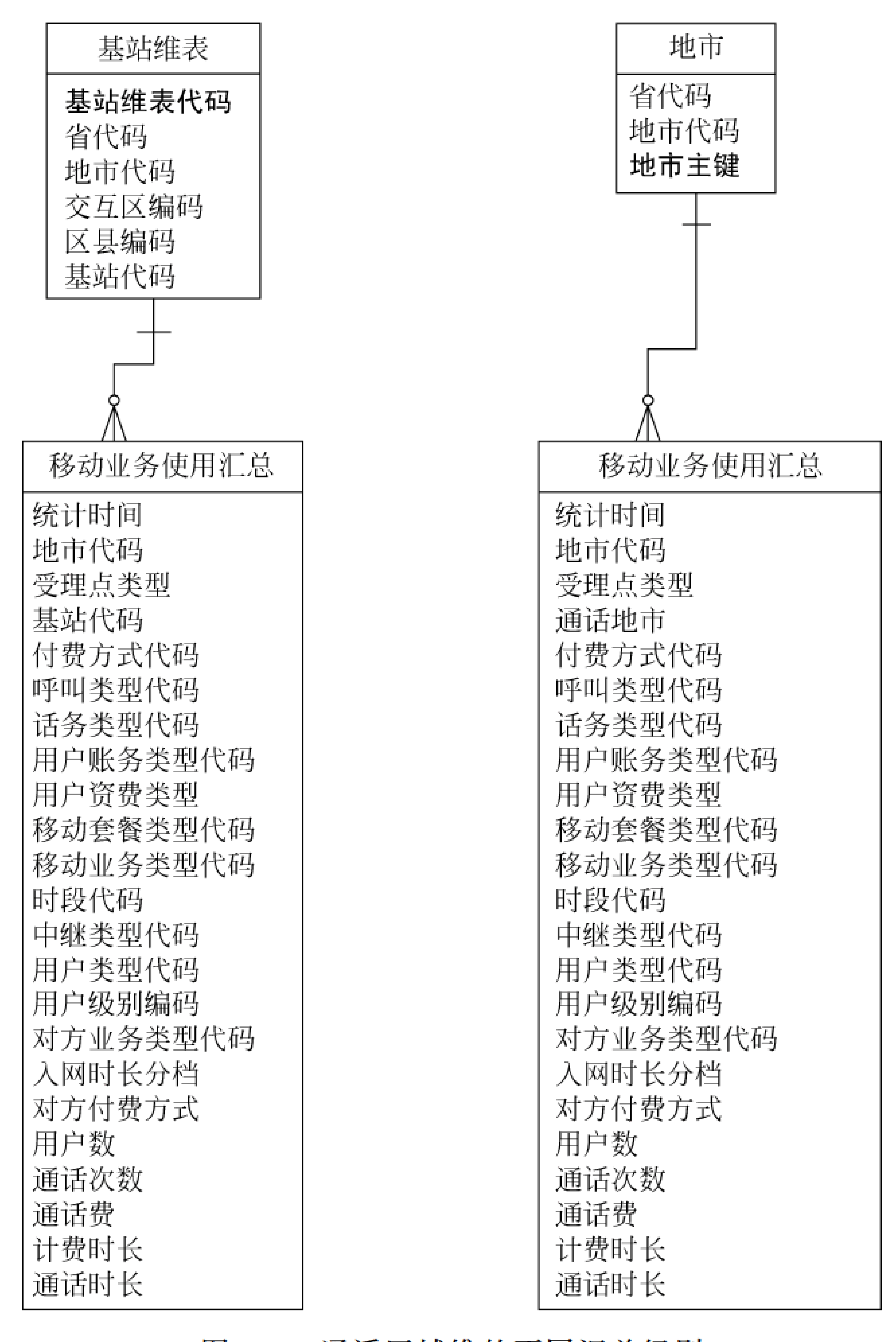


图 3.7　通话区域维的不同汇总级别

采用上述方式可以避免在同一维表中保存不同层次的代码，使维表结构清晰，同时减少单一维表的数据量。另外由于代码表的存在，使得不同粒度维表中同一层次的代码统一。

(3) 除了用户维表、客户维表等业务实体对应的维表之外，维表的层次字段使用统一的命名方式，分别为 LnCODE，其中 n 标识是第 n 层编码。采用统一的命名方式便于对维表的统一维护。大部分的维表变化是指增加相应的维度取值，对此类数据的变化主要是在代码表和维表中增加相应的代码，不会对历史数据产生影响。

4. 分割数据

数据分割是指首先根据业务系统的不同对表进行分割，例如按照 GSM、增值、数据和 VoIP 等不同的业务系统分割数据，这主要是基于各个不同的数据级别上都有业务维，同时不同业务除了一部分共性的数据外还包括属于该业务系统特殊的信息，因此按照不同业务分割数据一方面可以减少单表的数据量，同时可以保证数据仓库中各个业务信息的完整性。

按照业务分割数据后，由于话单和出账数据的单个业务的数据量还是很大，因此将话单和出账数据按照用户所属地域再次进行分割。选择用户所属地域是因为各个汇总级别上都包括用户所属地域维度，按照用户所属地域分割不会对数据抽取产生影响。

5. 划分表

对用户表按照其数据变化频率进行划分，大致可分为：

- 比较稳定的字段

如用户标识、用户号码、用户所属地域、入网时间、离网时间、用户类型和信用额度等。

- 经常变化的字段

如用户套餐、用户状态和停开机时间等。

由于用户套餐在用户查询中经常使用，同时由于是每天对业务系统的用户表进行抽取，因此对话务信息而言有可能丢失用户当时通话时的套餐信息，因此将用户套餐作为冗余字段放在各个汇总的事实表中。最终将用户信息划分成如下两张表。

- 用户基本信息表包括用户标识、用户号码、所属地域、入网时间、离网时间、用户类型、信用额度和用户套餐等字段。
- 用户在网状态表包括用户标识、用户套餐、用户状态和停开机时间等字段。

6. 设计多维模型

针对每个主题域确定其所需的维度和度量，然后定义每个主题域的关系模式，从而形成一个星型模式，在此基础上可以生成多维数据表，建立多维模型。

以客户主题域为例，其维度的设计如图 3.8 所示。

模型名称： 客户资料
模块功能： 用于客户数量的分析以及客户属性的分析
事实表： 客户资料事实表
度量： 客户数量
数据粒度：
每个客户每月计算一次收益，事实表中每条记录表示一个客户的属性。
事实表中存放 1 年以内的数据，超过 10 年的数据按月进行滚动，最初的数据汇总后从事实表中导出。
相关维度：
(1) 客户详细资料维
(2) 客户性别维
(3) 客户年龄层次维
(4) 客户在网时间维
(5) 客户消费层次维
(6) 客户信用度层次维
(7) 是否大客户维
(8) 付费类型维
(9) 地域维
(10) 客户流失概率层次维
(11) 客户挽留价值层次维
(12) 成为大客户概率层次维
(13) 客户价值层次维
(14) 客户服务状态维
(15) 客户号码维

图 3.8 客户主题域的维度设计

相应的星型模式如图 3.9 所示。

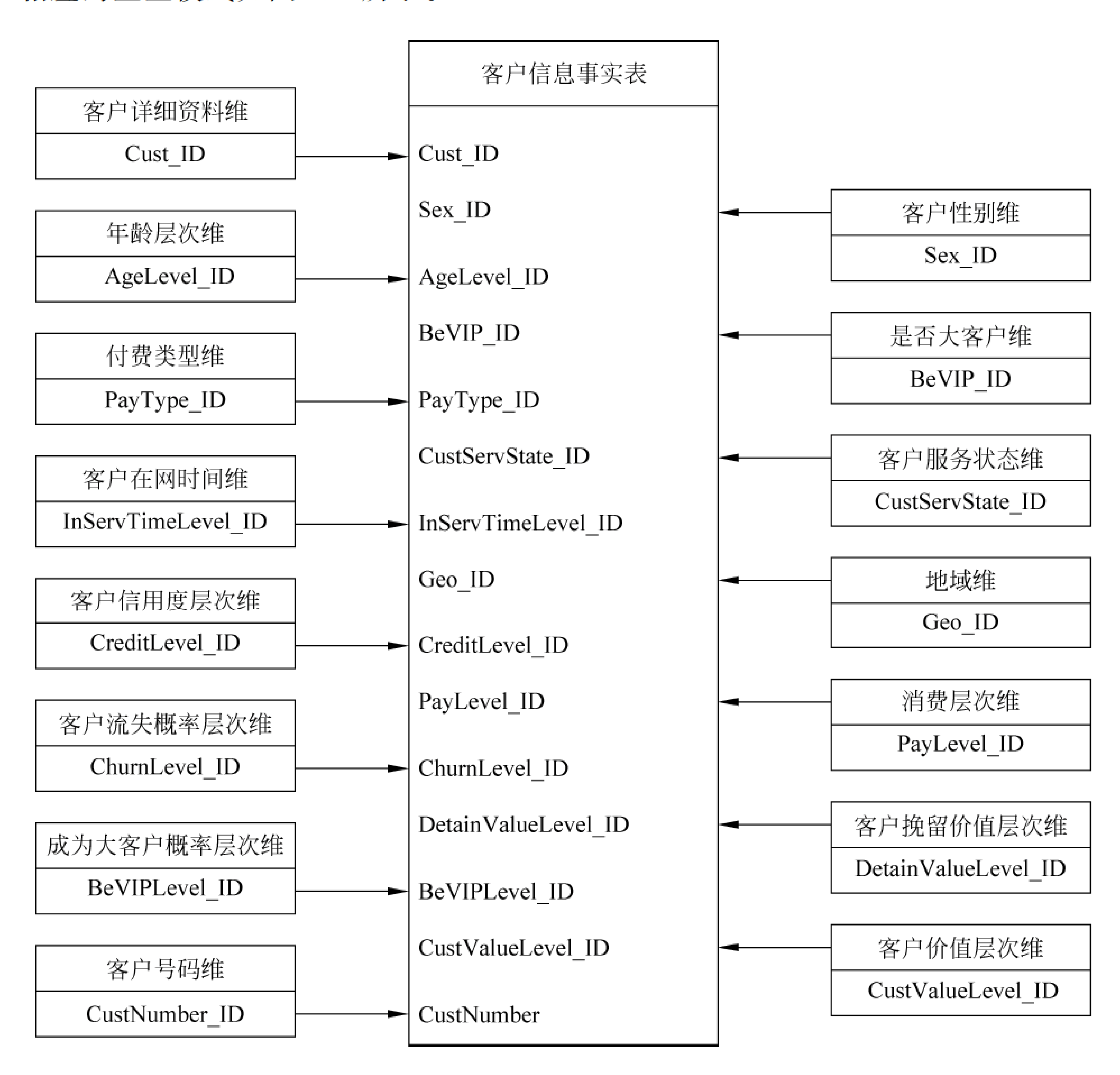


图 3.9　客户基本信息的星型模式

在关系数据库中实现多维模型即将多维模型中的度量、维度、事实和层次等概念用关系模型中的元素实现，其实现方式可以概括为：在关系模型中维转换成维表，事实转换成事实表，度量转换成事实表中的一个字段。维表和事实表的关联是通过将维表的主键作为事实表的外键实现。

在使用维表实现维时，需要对维的层次进行表示，具体如下：

（1）通过维表的不同字段

通过维表的不同列表示维的不同层次。例如，地域维中的层次为省、地市和区县，如图 3.10 所示。

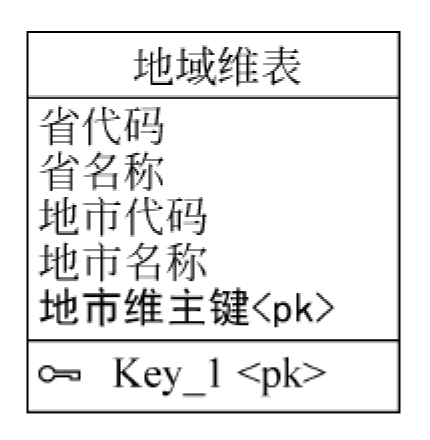


图 3.10　字段表示维的层次

（2）通过表之间的关联

使用不同的表分别表示维的不同层次，不同表之间通过外键关联形成维的层次。例如上述地域维的层次可以通过表之间的关联表示，如图 3.11 所示。

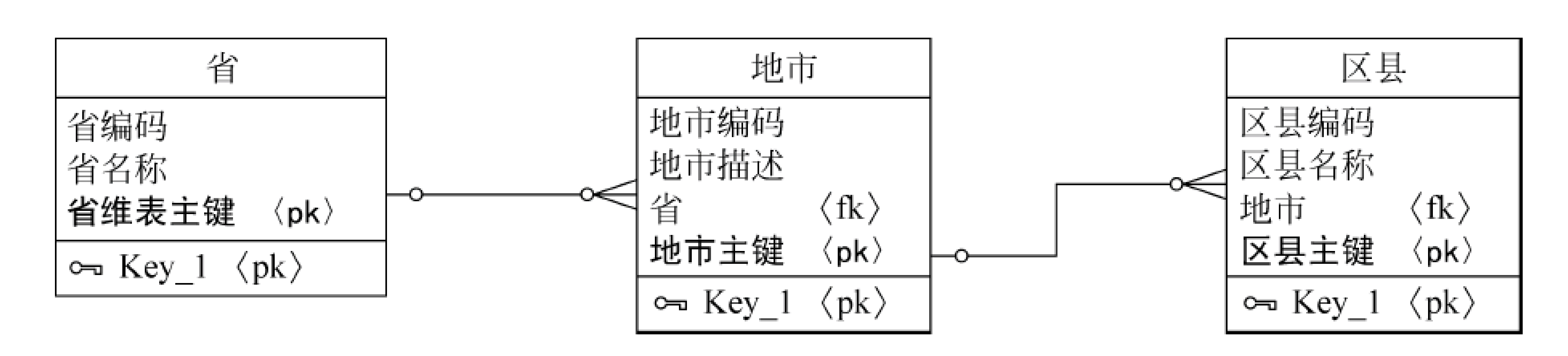


图 3.11 表之间的关联表示维的层次

(3) 通过关联字段

在维表中加入两个字段,一个标识维中的成员,另一个标识该成员的逻辑父代的成员,例如上述地域维的层次可以通过关联字段来表示,如图 3.12 所示。

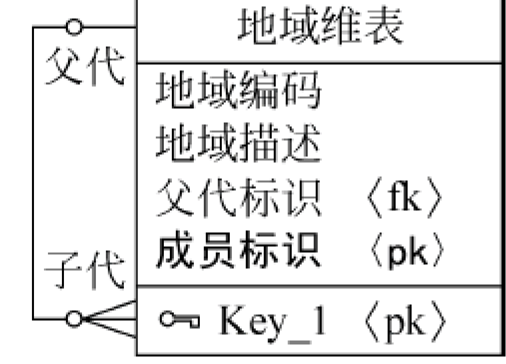


图 3.12 关联字段表示维的层次

上述各种维层次的表示方法中,通过字段表示的方法最为简单,而且由于查询时是对单表的查询,因此查询效率较高。通过不同的表来表示维的层次可以提取出许多共用的维表,提高维表的可维护性,但是由于查询时需要进行多次的关联,因此查询效率较低。以上两种方法对于不均衡和不整齐的维而言,都存在数据冗余,需要通过引入人工处理将维度补齐,如果维的层次较多,最终实现起来数据冗余会很大。而采用关联字段的方法,可以很好地表示不均衡和不整齐的维度,但是由于在查询时需要对表进行多次的自关联操作,因此查询效率较低,而且采用关联字段的维表也不易被用户理解。星型和雪花型模式中都可以使用关联字段实现维表。

3.1.3 物理模型

本例中,物理模型的设计主要包括以下几个方面。

1. 数据文件的存储分配

由于数据仓库中 ODS、轻度综合级、高度综合级以及维表的数据量和数据增长方式的不同,故将其分成多个数据文件进行存储,又因为话单的数据量很大同时数据增长也很快,因此在 ODS、轻度综合级和高度综合级对应的数据文件中都指定几个文件作为话单数据专用的数据文件。

2. 表的索引

当前细节级中的用户表、欠费表加入主键索引可以提高数据更新速度。对其他的表,可以根据用户经常查询的方式以及字段本身的特性加入适当的索引。

3. 表的物理分割

对话务和出账数据按照时间进行物理分割,使得对一天或者一个月数据的查询不受数据逐渐增加的影响。物理分割依赖于 DBMS 系统的功能,例如使用 Oracle 提供的分区功能对数据进行物理分割。

4. 禁止外键关联

将数据模型中的外键约束禁止变为人工约束,将外键禁止是为了保证数据抽取的效率,不删除外键是为了提供数据之间的关联关系。

3.1.4 ETL

本例中,ETL 设计主要包括以下几个方面。

1. 抽取数据到临时数据区

数据仓库的数据源不是来自终端客户每天输入的数据,而主要来自企业的 OLTP 系统的数据。

对于一个简单的面向电信领域的数据仓库,在将 OLTP 系统的数据加载到数据仓库之前,需先将其抽取到一个临时数据区,并在临时数据区进行数据清理和校验工作,待全部数据正确无误后,再加载到数据仓库,以保证数据质量。这是使用临时数据区的一个好处。

临时数据区事实上就是一个数据库,此数据库作为数据仓库的数据源,数据仓库直接从临时数据区加载数据,从而避开 OLTP 系统,这样避免了 OLTP 系统和数据仓库系统之间处理上的冲突,避免了 OLTP 系统因数据抽取而影响其响应时间。这是使用临时数据区的另一个好处。

临时数据区暂存将要加载到数据仓库的事实数据和维度数据,它利用关系表映射数据仓库的星型和雪花型模式。如果数据仓库建立在 SQL Server 2000 数据库基础上,则从 OLTP 系统抽取数据时,必然会用到存储过程和脚本文件。

下面以时间维数据、客户维数据和收益事实表数据的抽取为例加以说明。

在 SQL Server 2000 中建立新数据库,命名为临时数据区,分别建立客户维表、五一维表、十一维表和收益事实表,分别如表 3.4～表 3.7 所示。

表 3.4 客户维表

字段名	数据类型	长度	说明
客户序号	Int	4	关键字,自增
客户号	Char	4	外部关键字
客户姓名	Char	10	非空
客户性别	Char	2	非空

表 3.5 五一维表

字段名	数据类型	长度	说明
五一序号	Int	4	关键字,自增
年	Int	4	非空
日	Int	4	非空
日期时间	Datetime	8	非空

表 3.6 十一维表

字段名	数据类型	长度	说明
十一序号	Int	4	关键字,自增
年	Int	4	非空
日	Int	4	非空
日期时间	Datetime	8	非空

表 3.7　收益事实表

字　段　名	数 据 类 型	长　　度	说　　明
事实序号	Int	4	关键字,自增
客户序号	Int	4	外部关键字
消费金额	Money	8	非空
日期时间	Datetime	8	非空

实现数据抽取的存储过程如下：

```
Create Procedure 客户主题抽取 as    //从客户表中抽取客户维数据
Select OLTP..客户表.客户号, OLTP..客户表.客户姓名,
  Case OLTP..客户表.性别
    When'0'Then '男'
    When'1'Then '女'
    When'm'Then '男'
    When'f'Then '女'
    Else OLTP..客户表.性别
  End,
Into 客户维度表 From OLTP..客户表
//从收益表中提取五一维数据
Select Datepart(yyyy, OLTP..收益表.收益日期时间),
       Datepart(dd, OLTP..收益表.收益日期时间),
       OLTP..收益表.收益日期时间
Into 五一维表 From OLTP..收益表
Where Datepart(mm, OLTP..收益表.收益日期时间) = 5 And
      Datepart(dd, OLTP..收益表.收益日期时间)>= 1 And
      Datepart(dd, OLTP..收益表.收益日期时间)<= 7
//从收益表中提取十一维数据
Select Datepart(yyyy, OLTP..收益表.收益日期时间),
       Datepart(dd, OLTP..收益表.收益日期时间),
       OLTP..收益表.收益日期时间
Into 十一维表 From OLTP..收益表
Where Datepart(mm, OLTP..收益表.收益日期时间) = 10 And
      Datepart(dd,OLTP..收益表.收益日期时间)>= 1 And
      Datepart(dd, OLTP..收益表.收益日期时间)<= 7
//从收益表中提取事实数据
Select
  临时数据区..客户维度表.客户序号, OLTP..收益表.消费金额,
  OLTP..收益表.收益时间日期
Into 客户收益事实表 From 临时数据区..客户维度表, OLTP..收益表
Where 临时数据区..客户维度表.员工号 = OLTP..收益表.客户号
```

在转换过程中，假设 OLTP 系统可能的性别集合为{{男,女},{0,1},{m,f}},则可以建立如下脚本：

```
//*************************************************************
// Visual Basic 转换脚本
```

```
//**************************************************************
Function Main()
If (DTSSource("sex") == "0")
  DTSDestination("性别") = "男"
Else if (DTSSource("sex") == "1")
  DTSDestination("性别") = "女"
If (DTSSource("sex") == "m")
  DTSDestination("性别") = "男"
Else if (DTSSource("sex") == "f")
  DTSDestination("性别") = "女"
Main = DTSTransformStat_OK
End Function
```

例如,源数据库的客户表和数据仓库的客户维表的结构如表 3.8 所示。

表 3.8　源数据库的客户表和数据仓库的客户维表的结构

源数据库的客户表		数据仓库的客户维表	
客户识别码	Char	客户当前标志	Int(标识)
客户消费能力	SmallMoney(年收入)	客户识别码	Char
Insert_dtm	Smalldatetime	客户消费能力	Varchar(收入层次)
Update_dtm	Smalldatetime		

通过两者的比较发现客户消费能力的数据类型不一样。前者存储的是具体的收入值,后者是收入的层次,可编写如下的 VBScript 脚本完成这一转换工作。

```
//**********************************************************************
//Visual Basic 转换脚本
//**********************************************************************
//将客户表的 yearly_income 的 SmallMoney 型转换成客户维表的 Varchar 型
Function Main()
Select Case DTSSource("yearly_income")
      Case Is <= 10000
          DTSDestination("yearly_income") = "$10k-"
      Case 10001 to 30000
          DTSDestination("yearly_income") = "$10k-$30k"
      Case 30001 to 50000
          DTSDestination("yearly_income") = "$30k-$50k"
      Case 50001 to 70000
          DTSDestination("yearly_income") = "$50k-$70k"
      Case 70001 to 90000
          DTSDestination("yearly_income") = "$70k-$90k"
      Case 90001 to 110000
          DTSDestination("yearly_income") = "$90k-$110k"
      Case 110001 to 130000
          DTSDestination("yearly_income") = "$110k-$130k"
      Case 130001 to 150000
          DTSDestination("yearly_income") = "$130k-$150k"
```

```
        Case else
            DTSDestination("yearly_income") = " $ 150k + "
        Main = DTSTransformStat_OK
End Function
```

2. 加载数据到数据仓库

从数据源抽取数据到临时存储区后,需要加载数据到数据仓库,其具体步骤如图 3.13 所示。

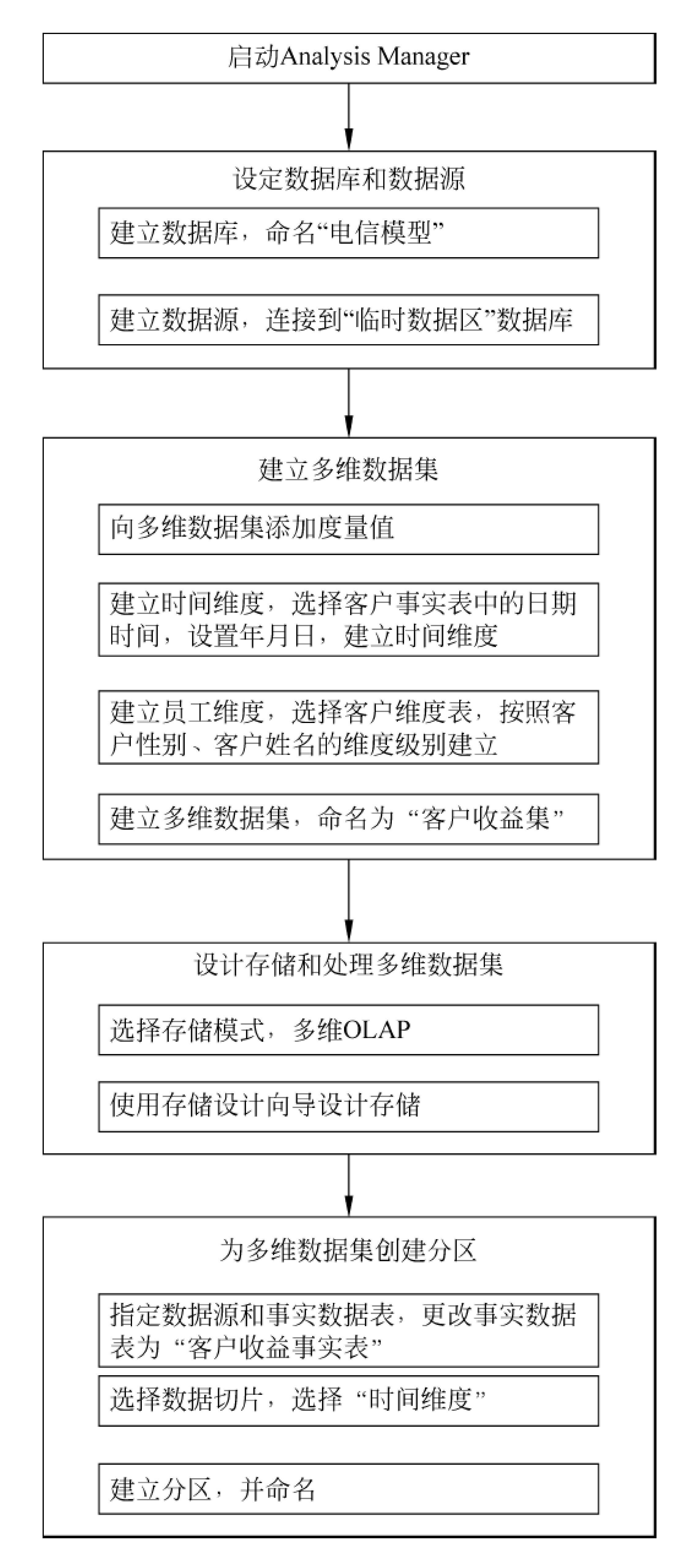


图 3.13　加载数据到数据仓库的具体步骤

3.2 实例二

随着数据仓库技术在金融服务业(如银行、保险等)、电信业、航空业等资金充足、信息化起步较早、迫切需要数据分析行业的成功应用。对其他领域的众多企业而言,如何在现有数据库基础上构建数据仓库日益迫切。在此以某公司的数据仓库建设为例,介绍数据仓库从设计到实现的完整过程。

3.2.1 总体结构

1. 设计原则

遵循统一平台架构、分阶段实施的原则,整个公司数据仓库和统计分析系统使用相同的软硬件平台,并结合业务的需求及业务现状,分领域、分阶段实施;保证系统在将来可以平滑地进行功能增加和规模扩充。在整个设计过程中,应遵循如下原则:

1) 系统实用原则,适应业务、技术的发展

为应对不断发展及变化的市场环境,公司业务结构、业务流程、产品开发和市场策略也需要相应地调整和优化,随之而来的是业务部门数据需求的调整和变化,相应的报表、查询、统计和分析也会发生变化。因此数据仓库和统计分析系统需要具有很强的业务适应能力,能够及时将业务变化反映在报表、查询和统计中。同时,随着新技术不断涌现,在系统的建设过程中也会出现新的概念和技术,系统在建设之前应充分考虑现有及未来技术的发展趋势。

2) 安全性和可靠性原则

可靠性方面,在业务系统数据正确完整的前提下,保证系统数据不丢失,不出现数据不一致;安全性方面,在不影响性能的同时,采用根据业务需要、授权最小的原则,确保系统的内部和外部安全。充分利用现有网络等环境资源,利用成熟的图形界面技术和经验,保证用户界面友好,易于使用、维护简单。为了保证进度、保证质量,利用成熟的工具,遵循软件工程的原则。

2. 总体结构设计

总体结构设计是对数据仓库系统的总体描述,它从宏观和整体的角度对数据仓库系统的各个组成部分进行总体设计,并确定在设计过程中遵循的总体原则,从而保证数据仓库各个组成部分在开发过程中能够依据同样的基础和标准,在运行过程中能够相互配合。本例中,采用建立 CIF 的数据仓库和 MD 的数据集市相结合的方法,以及平行开发模式搭建数据仓库,其总体结构如图 3.14 所示,分为数据整合层、数据服务层和信息展现层。

1) 数据整合层

该部分主要完成从 OLTP 系统(包含各类数据源,如 ERP、工程项目管理系统和业

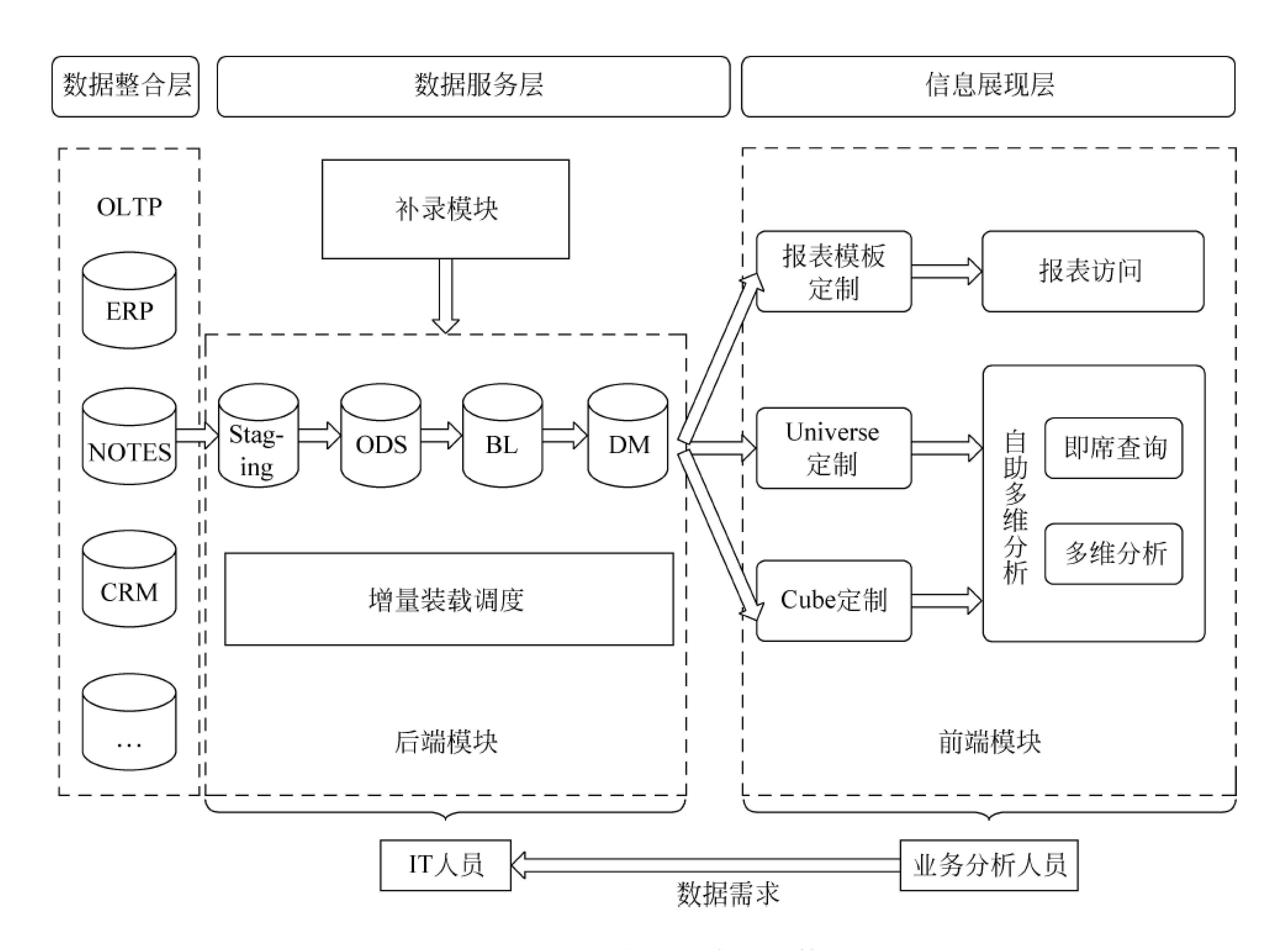


图 3.14　某公司数据仓库的总体结构

务管理系统等)通过 ETL 工具将数据载入数据仓库。某公司 ERP 采用的是 Oracle,即 OLTP 数据源为 Oracle,由于本例中数据仓库采用 Oracle 9i 作为数据库服务器,所以可以直接通过建立数据库连接的方式增量地从 OLTP 数据源中抽取数据(ETL 采用 Oracle 存储过程)。如果 OLTP 数据源不是 Oracle,则需通过 ODBC、LEI 等方式抽取数据。

2) 数据服务层

该部分主要完成数据仓库的数据分段存储。针对现有各业务系统进行数据抽取、清理并有效集成,按照主题进行组织,并建立针对主题的多维模型。考虑处理历史数据的效率及数据仓库远期规划等问题,业界通常将源系统加载到数据仓库中进行数据分层存储。在本例中,将数据服务层的数据存储划分为如下区域:

- 临时存储区(Staging Area)是为了保证数据迁移的顺利进行而设置的增量式的数据存储空间,它使业务系统原始数据进入数据仓库的缓冲区。需要进入数据仓库各主题域的数据首先直接快速传输到临时存储区,然后再转移到目标数据仓库中。从业务系统(如 ERP、PSP、Notes)到临时存储区的传输应避免进行复杂的数据处理,以保证数据的快速导入而减少对业务系统造成的压力。一般地,可以创建与 OLTP 系统实体结构相同的属性,同时在临时存储区中增加两个属性:

(1) Source code 用来表示来源系统。

(2) Last Modified Date 用来获得数据处理时间。如果原来的数据中已经具有上述属性，则需要在新属性中增加 dw 后缀进行标识；数据成功导入数据仓库后，应清空临时存储区的数据。

- 操作数据存储区(Operational Data Store，ODS)是为了保证数据迁移的顺利进行而设置的数据存储空间，需要进入数据仓库各个主题域的数据从临时存储区直接快速传输到 ODS，再从 ODS 经过清洗、转换、映像等复杂的数据处理载入目标数据仓库中。ODS 的数据作为数据仓库系统数据存储。逻辑上，ODS 可以分为两部分：一部分存放 OLTP 系统的历史数据，这部分需要重新考虑是否需要对 OLTP 的数据进行生命周期的记录(包括交易数据与基础数据，即缓慢变化的处理)；另一部分存放数据仓库加工的信息，即 ODS 历史数据经过整合后的信息，这些信息更加全面地反映一个主题域中某一事物的全貌。
- 中央数据仓库(BaseLine)是具有星型或雪花型结构的实体，包括事实实体(Fact Entity)和维度实体(Dimension Entity)。其中，事实实体是对某一个事物(可能是某笔交易、某个项目、某笔到货明细，某个任务)各个方面信息的全面描述，描述的属性包括该事物各方面的度量信息，相关度量信息的相关维度信息；维度实体，此处的维度是与事实实体相关的维度信息，包括很多事实实体共有的维度信息，如时间维等，以及某一个 Fact Entity 专有的维度信息。Baseline 需要能够支持最细粒度级，保证可以在最细粒度级实现多维分析，即能够同时支持汇总和细节数据的多维查询。
- 数据集市(Data Mart)是某一主题域的专有多维数据区，实现某一主题域的多维查询。这一部分也包括事实实体和维度实体两种实体，但与 Baseline 不同的是数据集市的事实实体和维度实体都是为某一业务主题服务的。

3) 信息展现层

信息展现层是指采用不同形式连接企业数据仓库，抽取不同的数据，主要包括即席查询、统计报表等。本例中，即席查询、统计报表采用 BusinessObjects 和 Web Intelligence 作为客户端，可以灵活地进行钻取、切片和旋转等多维分析操作。

3.2.2 概念模型

数据仓库概念模型设计的目的是对数据仓库所涉及的现实世界中所有客观实体进行科学、全面地分析和抽象，为构建数据仓库制定“蓝图”。在概念模型中，常用 E-R 图作为描述工具。在 E-R 图中，长方体表示实体，即表示数据仓库中的主题域，框内写上主题域的名称；椭圆表示主题域的属性，用无向边把主题域与其属性连接起来，有向边表示主题域之间的联系(单向边表示一对多的关系，双向边表示多对多的关系)，无向边表示主题域之间一对一的关系。在此，以质量绩效分析主题为例进行概念模型设计。

质量绩效分析主要通过产品族、部门等不同角度对公司产品质量情况进行分析，涉及的主要绩效指标(Key Performance Indication，KPI)包括紧急版本发布比例、版本测试不通过

率、网上客户解决率、研发网上问题解决率和保修期内产品故障率等。质量绩效分析的概念模型如图 3.15 所示。

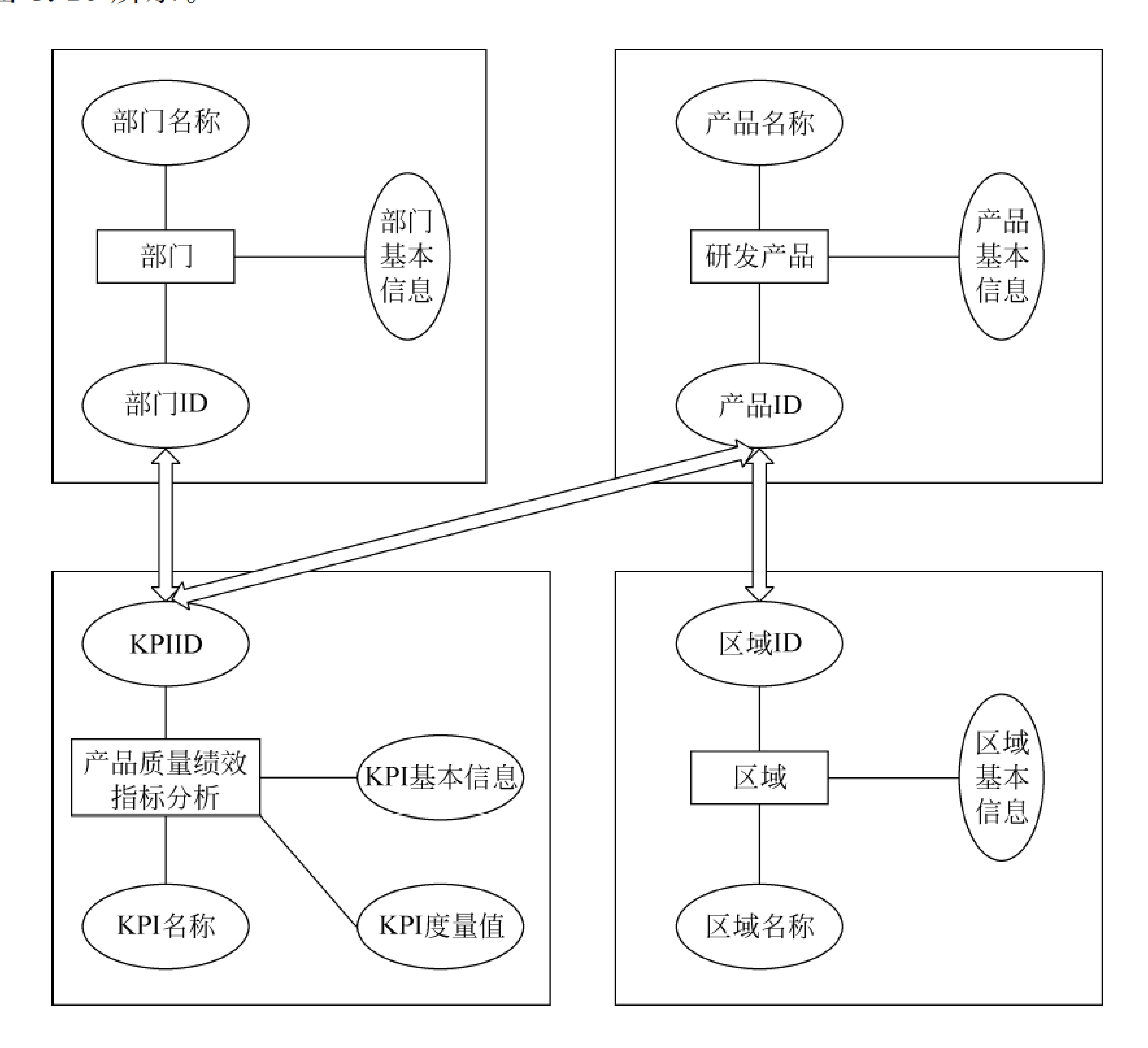


图 3.15　质量绩效分析的概念模型

3.2.3　逻辑模型

逻辑模型设计是对概念模型设计中确定的基本主题域进行分析,并详尽定义。本例中,逻辑模型的设计采用多维模型,根据具体业务的需要,设计为星型、雪花型和星型-雪花型等模式。

1. 多维模型设计

质量绩效指标涉及相同的维度信息,并且计算方法相同,所以可设计相同的事实表实现对不同绩效指标的分析。由于质量绩效的研发产品维信息涉及产品线维、产品族维、产品开发团队维和测评产品维,所以质量绩效分析主题的逻辑模型采用雪花型,如图 3.16 所示。

1) 时间维表

几乎所有的数据仓库都会包括时间维表,时间维相对独立,变化较少。本例中时间维表的结构如表 3.9 所示。

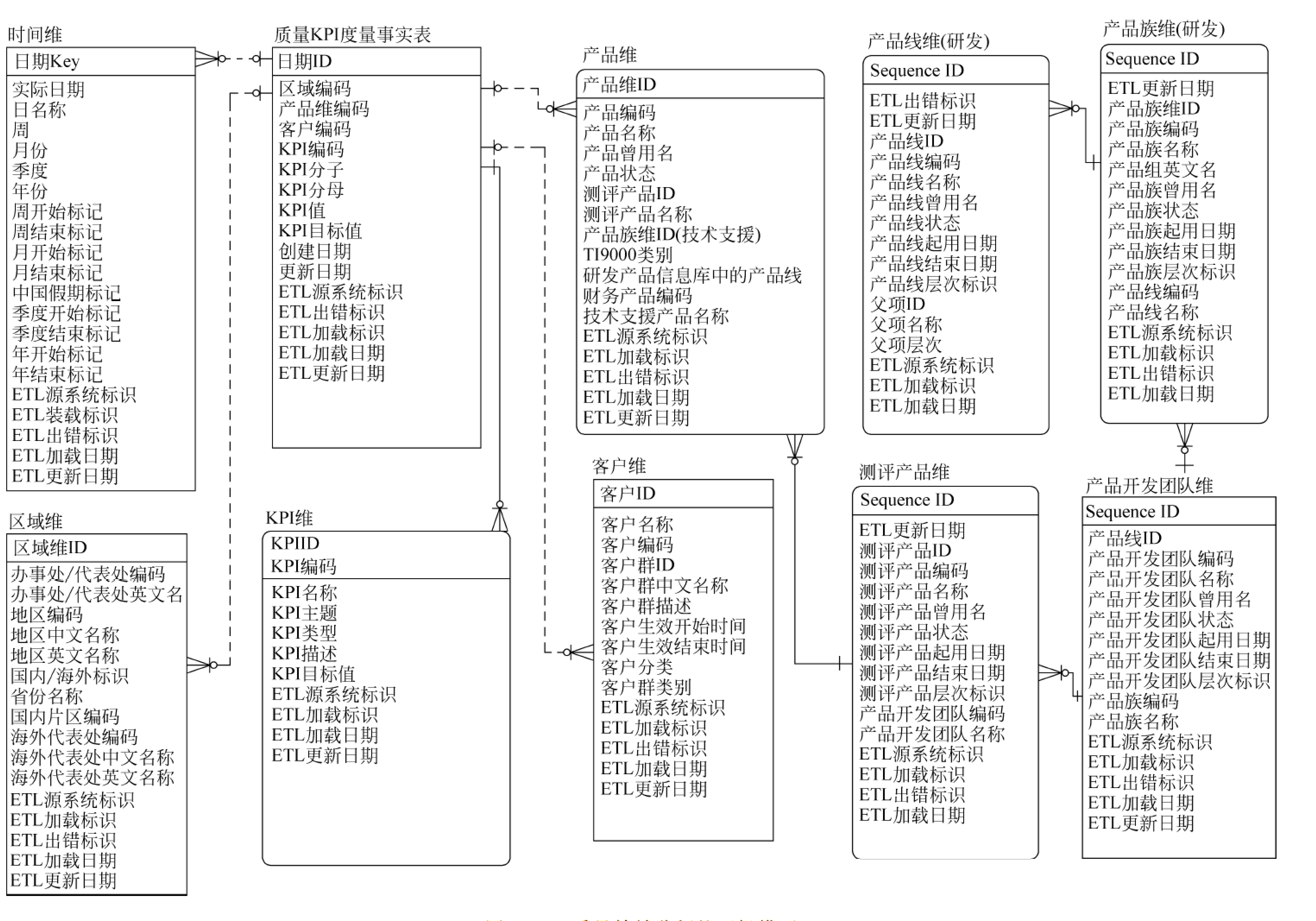


图 3.16　质量绩效分析的逻辑模型

表 3.9　时间维表

列　　名	中 文 名 称	类　　型	备　　注
Date_key	日期键	Number	Not null
Actual_date	实际日期	Datetime	Not null
Week	周	Number	Not null
Month	月	Number	Not null
Quarter	季度	Number	Not null
Year	年	Number	Not null
Effective	状态(有效 1,停用 0)	Number	Not null
Start_Week_mark	周开始标记	Varchar2(1)	Not null
End_Week_mark	周结束标记	Varchar2(1)	Not null
Start_Month_mark	月开始标记	Varchar2(1)	Not null
End_Month_mark	月结束标记	Varchar2(1)	Not null
Start_Year_mark	年开始标记	Varchar2(1)	Not null
End_Year_mark	年结束标记	Varchar2(1)	Not null
ETL_soure_code	源系统标识	Varchar2(50)	Null
ETL_load_mark	ETL 加载标识	Varchar2(20)	Null
ETL_error_code	ETL 出错代码	Varchar2(20)	Null
ETL_loading_date	ETL 加载日期	Date	Null
ETL_update_date	ETL 更新日期	Date	Null

2）研发产品线维表

研发产品线维表也是大多数数据仓库都需要用到的基本维表之一，本例中研发产品线维表的结构如表 3.10 所示。

表 3.10　研发产品线维表

列　　名	中 文 名 称	类　　型	备　　注
Seq_id	序列(代理键)	Int	Not null
Product_line_dev_id	研发产品线维 ID	Number	Not null
Product_line_dev_code	产品线编码	Varchar2(20)	Not null
Product_line_name	产品线名称	Varchar2(50)	Not null
Product_line_pre	产品线曾用名	Varchar2(50)	Not null
Effective	状态(有效 1,停用 0)	Number	Not null
Start_date	起用日期	Date	Not null
End_date	结束日期	Date	Null
Level_flag	层次标识	Int	Not null
Par_id	父项 ID	Int	Not null
Par_name	父项名称	Varchar2(50)	Not null
Par_level	父项层次	Int	Not null
ETL_soure_code	源系统标识	Varchar2(50)	Null
ETL_load_mark	ETL 加载标识	Varchar2(20)	Null
ETL_error_code	ETL 出错代码	Varchar2(20)	Null
ETL_loading_date	ETL 加载日期	Date	Null
ETL_update_date	ETL 更新日期	Date	Null

值得注意的是：

(1) 研发产品线维表中缓慢变化的为产品线编码和产品线名称，即随着时间的变化产品线编码和产品线名称可能发生变化。如某固网产品的产品线，变化成终端固网产品线，所以应保存原来的产品信息，还要增加新的产品信息。具体处理如下：在维表加载过程通过研发产品线维 ID 检测已存在该产品线但产品线名称发生变化，可采用将原来的产品线标记无效，即 Effective＝0，同时修改 End_date 为当前日期，新增一条新的记录，即 Start_date＝当前日期、End_date＝NULL、Effective＝1。这样可以保留原来的产品线信息，同时又增加新的产品线信息。

(2) 所有公共维表应存储在 Baseline 层。

3) 研发产品族维表

研发产品族维表和研发产品线维表的关系为典型的子表和主表的关系。考虑维表维护的简单性和加载的便捷性，对研发产品族维表和研发产品线维表单独设计，而不是合并在同一个维表中，本例中研发产品族维表的结构如表 3.11 所示。

表 3.11　研发产品族维表

列　名	中文名称	类　型	备　注
Seq_id	序列	Int	Not null
Prodfamily_dev_id	研发产品线维 ID	Number	Not null
Prodfamily_code	产品族编码	Varchar2(20)	Not null
Prodfamily_name	产品族名称	Varchar2(50)	Not null
Prodfamily_pre	产品族曾用名	Varchar2(50)	Not null
Prodfamily_name_en	产品族英文名	Varchar2(50)	Not null
Effective	状态(有效 1,停用 0)	Number	Not null
Start_date	起用日期	Date	Not null
End_date	结束日期	Date	Null
Level_flag	层次标识	Int	Not null
Product_line_dev_id	研发产品线维 ID	Int	Not null
Product_line_name	研发产品线名称	Varchar2(50)	Not null
ETL_soure_code	源系统标识	Varchar2(50)	Null
ETL_load_mark	ETL 加载标识	Varchar2(20)	Null
ETL_error_code	ETL 出错代码	Varchar2(20)	Null
ETL_loading_date	ETL 加载日期	Date	Null
ETL_update_date	ETL 更新日期	Date	Null

值得注意的是：研发产品族维表中缓慢变化的为产品族编码和产品族名称，其处理方法和研发产品线维相似。其中 Product_line_dev_id 和 Product_line_name 为该产品族的产品线信息，这样可以很方便地处理某一产品族的产品线信息，在前台 OLAP 报表可以很方便地在研发产品线维和研发产品族维之间进行向上和向下钻取。

4) 其他维表设计

其他维表的设计可以参照研发产品线维表和研发产品族维表，不再赘述。

5) 事实表设计

在本例中，事实表主要分两类：一类是保存在 Baseline 层(中央数据仓库)的事实表，这

类事实表存放细节数据；另一类是保存在 Data Mart 层的事实表，这类事实表存放某一部门或某一领域内的汇总数据。Baseline 层和 Data Mart 层事实表的物理结构有些是一样的，唯一区别是物理数据是分层存放的。就数据粒度而言，Baseline 层和 Data Mart 层的事实表的数据粒度也是不同的。Baseline 层和 Data Mart 层的事实表在物理结构上有的是完全不同的，因为 Data Mart 层数据可以直接来源于 ODS 层，而不是 Baseline 层；就数据集市的类型而言，数据集市可以是独立的和从属的数据集市。本例中整个数据仓库的架构非常灵活，综合了 CIF 数据仓库和 MD 数据集市的优点，可满足企业建立独立的数据集市和从属的数据集市的需求。所以事实表的设计将从 Baseline 层和 Data Mart 层两方面展开。

质量绩效分析事实表的结构如表 3.12 所示。

表 3.12 质量绩效分析事实表

列 名	中文名称	类 型	备 注
Seq_id	序列	Number	Not null
Date_id	日期 ID	Number	Not null
Region_id	区域 ID	Number	Not null
Product_id	研发产品 ID	Number	Not null
Department_id	部门 ID	Number	Not null
KPI_code	KPI 编码	Varchar2(10)	Not null
Prodfamily_name_en	产品族英文名	Varchar2(50)	Not null
YTD_KPI_valuenumber	上年同期 KPI 值	Number	
KPI_value_number	KPI 值	Number	Not null
Period_level	日期层次	Varchar2(10)	Not null
KPI_type	KPI 类型	Varchar2(20)	Not null
KPI_Factor1	KPI 分子 1	Number	Not null
KPI_Factor2	KPI 分子 2	Number	Not null
KPI_Factor3	KPI 分母 1	Number	Not null
KPI_Factor4	KPI 分母 2	Number	Not null
YTD_KPI_factor1	上年同期 KPI 分子 1	Number	Not null
YTD_KPI_factor2	上年同期 KPI 分子 2	Number	Not null
YTD_KPI_factor3	上年同期 KPI 分母 1	Number	Not null
YTD_KPI_factor4	上年同期 KPI 分母 2	Number	Not null
ETL_soure_code	源系统标识	Varchar2(50)	Null
ETL_load_mark	ETL 加载标识	Varchar2(20)	Null
ETL_error_code	ETL 出错代码	Varchar2(20)	Null
ETL_loading_date	ETL 加载日期	Date	Null
ETL_update_date	ETL 更新日期	Date	Null

该事实表在 Baseline 层和 Data Mart 层可保持相同的物理结构，但 Data Mart 层数据是对 Baseline 层数据的聚合和汇总。

6）ODS 层数据结构

ODS 层数据结构如表 3.13 所示。

表 3.13 ODS 层数据结构

ODS 表名称	源 表 名 称
ODS_EXP_CONSIGNMENT_ORDER	EXP_CONSIGNMENT_ORDER
ODS_EXP_TRAFFIC_PLAN	EXP_TRAFFIC_PLAN
ODS_EXP_TRAFFIC_PLAN_ODER	EXP_TRAFFIC_PLAN_ODER
ODS_RCV_TRANSACTIONS	RCV_TRANSACTIONS
ODS_RCV_SHIPMENT_LINES	RCV_SHIPMENT_LINES
ODS_RCV_SHIPMENT_HEADERS	RCV_SHIPMENT_HEADERS
ODS_PO_VENDORS	PO_VENDORS
ODS_PO_HEADERS_ALL	PO_HEADERS_ALL
ODS_MTL_SYSTEM_ITEM_B	MTL_SYSTEM_ITEM_B
ODS_HR_ALL_ORG_UNITS	HR_ALL_ORG_UNITS
ODS_IBS_HKBOXLIST_DETAIL	IBS_HKBOXLIST_DETAIL
ODS_IBS_HKSENDLIST_MASTER	IBS_HKSENDLIST_MASTER
ODS_IBS_HKSENDLIST_DETAIL	IBS_HKSENDLIST_DETAIL
ODS_IBS_CORPINF	IBS_CORPINF
ODS_IBS_TRUCKCODE	IBS_TRUCKCODE
ODS_IBS_HKSHIPREC_DETAIL	IBS_HKSHIPREC_DETAIL
ODS_T_PUB_DEPT	T_PUB_DEPT
ODS_T_PRODUCT	T_PRODUCT
ODS_T_REGION	T_REGION
ODS_T_ORDERS	T_ORDERS
ODS_T_ACP_INFO	T_ACP_INFO

7）临时存储层数据结构

临时存储层数据结构如表 3.14 所示。

表 3.14 临时存储层数据结构

ODS 表名称	取数方式	源系统名称	源表名称
EXP_CONSIGNMENT_ORDER	每天增量	EBS	EXP_CONSIGNMENT_ORDER
EXP_TRAFFIC_PLAN	每天增量	EBS	EXP_TRAFFIC_PLAN
EXP_TRAFFIC_PLAN_ODER	每天增量	ERP	EXP_TRAFFIC_PLAN_ODER
RCV_TRANSACTIONS	每天增量	ERP	RCV_TRANSACTIONS
RCV_SHIPMENT_LINES	每天增量	ERP	RCV_SHIPMENT_LINES
RCV_SHIPMENT_HEADERS	每天增量	ERP	RCV_SHIPMENT_HEADERS
PO_VENDORS	每天增量	ERP	PO_VENDORS
PO_HEADERS_ALL	每天增量	ERP	PO_HEADERS_ALL
MTL_SYSTEM_ITEM_B	每天增量	ERP	MTL_SYSTEM_ITEM_B
HR_ALL_ORG_UNITS	每天增量	ERP	HR_ALL_ORG_UNITS
IBS_HKBOXLIST_DETAIL	每天增量	ERP	IBS_HKBOXLIST_DETAIL
IBS_HKSENDLIST_MASTER	每天增量	ERP	IBS_HKSENDLIST_MASTER
IBS_HKSENDLIST_DETAIL	每天增量	ERP	IBS_HKSENDLIST_DETAIL
IBS_CORPINF	每天增量	ERP	IBS_CORPINF
IBS_TRUCKCODE	每天增量	ERP	IBS_TRUCKCODE

续表

ODS 表名称	取数方式	源系统名称	源表名称
IBS_HKSHIPREC_DETAIL	每天增量	ERP	IBS_HKSHIPREC_DETAIL
T_PUB_DEPT	每天增量	ERP	T_PUB_DEPT
T_PRODUCT	每天增量	ERP	T_PRODUCT
T_REGION	每天增量	ERP	T_REGION
T_ORDERS	每天增量	ERP	T_ORDERS
T_ACP_INFO	每天增量	ERP	T_ACP_INFO

2. 数据粒度设计

本例中 Baseline 层和 Data Mart 层采用的是多维模型,所以数据粒度设计也是针对 Baseline 层和 Data M art 层。Baseline 层保存企业业务的明细数据,该层的数据粒度为低粒度级(高细节级);Data Mart 层保存企业某一部门或某一主题的汇总或聚合数据,该层的数据粒度为高粒度级(低细节级)。

W. H. Immon 指出不同数量级采用的数据粒度策略如表 2.1 所示,在此数据粒度的选择借鉴了其数据粒度设计策略。

质量绩效分析事实表的数据在 5 年内可膨胀到 20 000 000,所以可考虑采用双重数据粒度,在 Baseline 层的事实表超过 5 年的细节数据可导出到后备存储设备,使 Baseline 层保存 5 年内的明细数据。其他维表的数据增量不大,可采用单一粒度。Baseline 层绩效分析事实表的数据粒度要达到产品层,Data Mart 层绩效分析事实表的数据粒度达到产品族层即可。

3. 分区设计

本例中,整个数据仓库采用分层存储。物理上分为四层,即 ODS 层、数据仓库层、数据仓库聚合层和数据集市层。由于目前的数据量不是很大,因此还不需要分区。但随着数据量的增加和数据仓库的扩展,数据分区是必然的,如可以按照时间和地域进行分区。

3.2.4 物理模型

本例中,数据仓库物理模型设计是在 Oracle 9i 数据库基础上展开的。

1. 设计原则

1) 表结构设计原则

(1) 列的数据类型

- 数字类型建议使用 Number 类型,一般不推荐使用 Integer 或 Float 类型。
- 如果类型为字符类型,一般不要使用 Char 类型,建议使用 Varchar2。
- 一般情况下,尽量避免使用 Type、LONG、BLOB 和 CLOB 等类型。
- 如果源系统为非 Oracle 数据库,如果数据类型在 Oracle 中没有直接对应的数据类型,参考 Oracle 手册选择相近的数据类型。

(2) 列的长度

如果源系统的字符集和数据仓库的字符集不一致,需要考虑调整 Varchar2 长度。如果源系统的中文存储方式为两个字节存储一个中文字符,在数据仓库中表的长度应该扩大(建

议 3 倍)。如果扩大 3 倍后超过 4000 字节,统一定义为 4000。

(3) PCTUSED 和 PCTFREE 参数

根据不同的要求,表可以分为三类:基本没有更新、少量更新和大量更新。临时存储区的表基本属于没有更新的类型,每次插入新数据先 Truncate 表,不做更新和删除操作。ODS、Baseline 层的表大部分属于少量更新的类型,以 Insert 为主,少量更新。Data Mart 层的表属于大量更新的类型,主要是汇总、更新和删除操作比较频繁。根据不同类型,PCTUSED 和 PCTFREE 参数建议设置如下:

基本没有更新 PCTUSED=90,PCTFREE=0

少量更新 PCTUSED=80,PCTFREE=10

大量更新 PCTUSED=70,PCTFREE=20

(4) PARALLEL 参数

为了提高性能,可以考虑使用 PARALLEL 参数,语法为 PARALLEL(DEGREE n),n 应该与表空间的数据文件数量一致,如果表空间的数据文件为 5 个,设置为 PARALLEL (DEGREE 5)。由于使用并发方式需要使用较多的系统资源,建议在需要提高性能时使用,一般情况不建议使用,设置为 NO PARALLEL 即可。

(5) LOGGING 参数

数据仓库中一般不需要进行 Log 处理,设置为 NO LOGGING 即可。

(6) 键设置

在数据仓库表的设计中,不建议为表创建主键或外键,如果需要进行约束,使用唯一索引或程序逻辑等方式代替主键或外键。如果必须创建主键,在定义主键时,先创建唯一索引,再创建主键,不要直接创建主键。

(7) 表空间参数

表空间参数务必和 Schema 的默认表空间一致,且不要与索引公用表空间。

(8) 表存储参数

表的初始大小建议设置为预计大小的 1/5~1/3,扩展参数可以不设置。

2) 表物理设计原则

表物理设计原则如表 3.15 所示。

表 3.15 表物理设计原则

设计项	内容
索引 PCTFREE 参数	索引的 PCTFREE 建议为 30
索引 PARALLEL 和 LOGGING	索引 PARALLEL 和 LOGGING 和表的相关参数设置原则相同
索引存储参数	建议初始设置为预计大小的 1/3
位图索引	在数据仓库中,有时需要设置位图索引,一般不建议使用,使用前先综合评估影响
创建脚本	表和索引的创建脚本必须包括 Schema,指定完整参数,不要遗漏表空间参数

2. 物理模型设计

1) 临时存储区的物理模型设计

(1) Schema list

(2) 表结构和索引

临时存储区中表结构基本和源系统对应表一致,表名一般相同,可参考逻辑模型设计中 Staging 层数据结构。一般情况下临时存储区中的表不使用索引。

2) ODS 层物理模型设计

(1) Schema list

(2) 表结构和索引

ODS 的表结构和临时存储区中表结构基本一致,可参考逻辑模型设计中 ODS 的数据结构。

3) Baseline 层物理模型设计

(1) Schema list

(2) 表结构和索引

对应缓慢变化维,要为所有缓慢变化的列建立唯一索引。

4) Data Mart 层物理模型设计

(1) Schema list

(2) KPI 事实表结构和索引

数据集市中 KPI 事实表由于更新较多,PCTUSED=70,索引 PCTUSED 要求与表的一致。

3.2.5 数据清洗

为保证数据仓库的数据质量,对正式进入数据仓库的数据必须采用有效的方式进行检查。如前所述,通常数据清洗可购买专用工具,也可以通过编写程序实现。本例中,采用 PL/SQL 编程对 ODS 层的数据进行清洗,由于数据仓库中数据质量涉及方方面面的问题,在此着重阐述接口数据的检查,从而解决数据仓库"垃圾进,垃圾出"的问题。

1. 清洗内容

本例中数据源主要来自 Oracle ERP 系统和其他管理信息系统,涉及的清洗内容包括:

- 实体完整性检查。如研发产品维中的产品代码不能为空,产品的状态只能为 1 或 0,不能为空等。
- 参照完整性检查(Referential Integrity)。如研发产品族维表中的产品线编码是研发产品线维表的外键。
- 业务规则的检查。如产品返回率的返还日期必须大于产品的发货日期。

2. 清洗规则设计

数据清洗逻辑模型如图 3.17 所示,其中表 CHK_RULE_DEF、表 CHK_TABLE_RULE_DEF、表 CHK_FIELD_RULE_DEF 和表 CHK_REFERENCE_KEY 用来保存数据清洗所用元数据,数据清洗元数据是在程序中创建初始脚本实现的。通过元数据定义可以灵活驱动多种数据清洗规则。表 CHK_CHECK_STATUS、表 CHK_EXCEP_LOG 和表 CHK_ERROR_LOG 用来保存数据清洗程序执行的状态和执行每项清洗规则的检查结果。

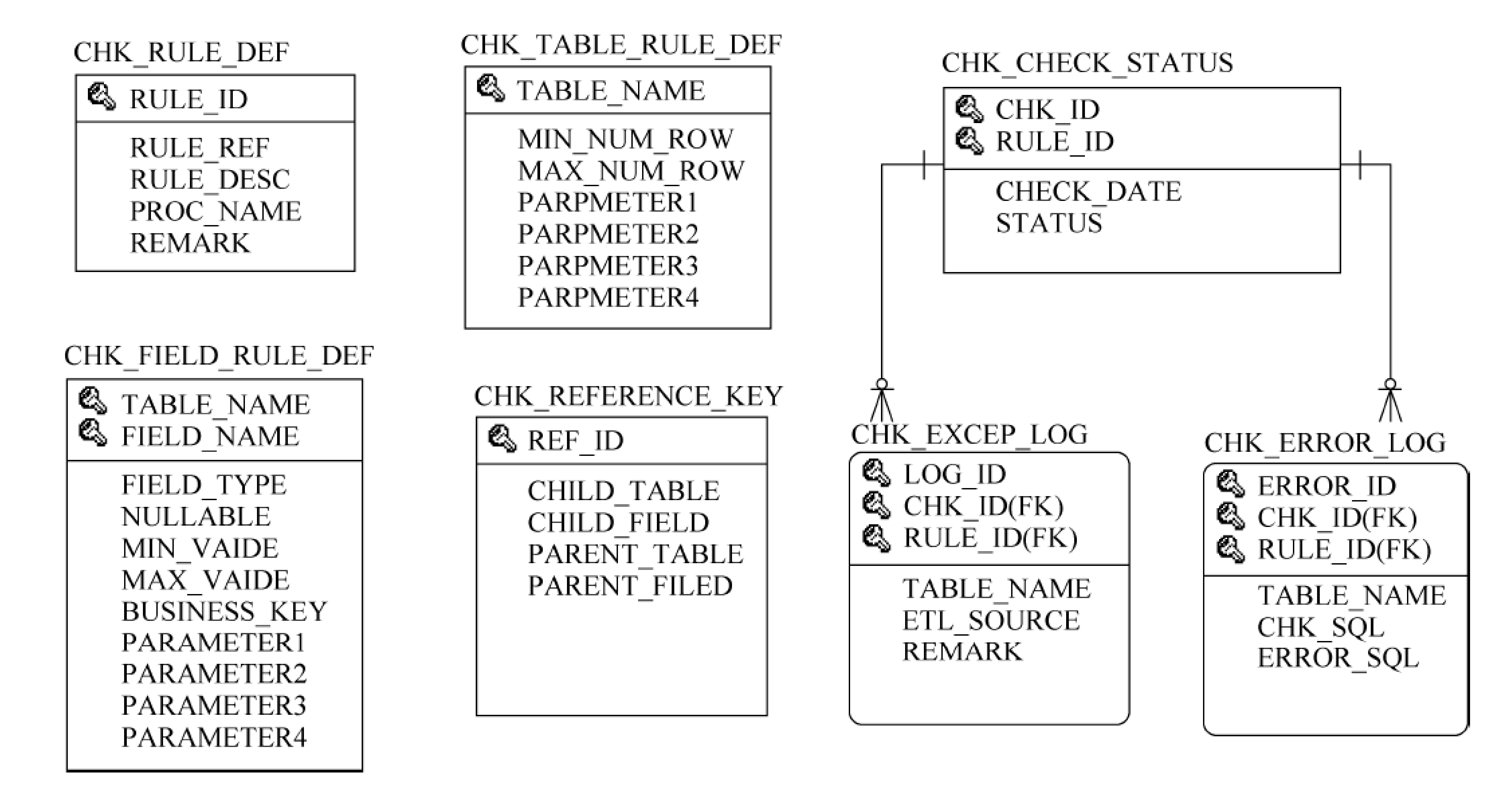


图 3.17　数据清洗逻辑模型

3.2.6　ETL

完成上述设计，即完成了搭建存储企业数据的“仓库”，但真正发挥所搭建“仓库”的作用，必须为其装入大量有价值的数据。通常数据源的数据格式存在很大的差异，为了保证数据仓库的数据完整性，ETL 实现数据的抽取、转换、清洁和装载，最终利用分析工具实现分析，支持企业经营决策。

本例中，ETL 工具采用 CA 公司的 Advantage Transformer Script Manager 2.0（简称 ADT），数据库平台为 Oracle 9i。

1. 数据抽取

从外部源系统加载数据到数据仓库的临时存储区，可以通过 SQL * Loader、外部表、导入导出、PL/SQL 等方式加载。如将表 EXP_CONSIGNMENT_ORDER（源系统为 Oracle ERP）加载到临时存储区，如果源数据量小于 1GB 实现代码如下：

```
procedure  sp_ exp_consignment_order
as
   Lv_program_name   varchar2(30) : = 'sp_exp_consignment_order';
   Lv_table_name     varchar2(32) : = 'exp_consignment_order';
   Lv_key_name1      varchar2(30) : = 'ID';
   Lv_key_name2      varchar2(30) : =  'Language';
   Lv_key_information   varchar2(2000) :  = null;
   Ln_key1 exp_consignment_order. id % type;
   Ln_key2 exp_consignment_order. Language' % type;
   Ln_ins_counter number: = 0;
   Ln_upd_counter number: = 0;
   Ld_start_date   date;
```

```
        Ld_end_date     date;
        Ln_commit_label smallint : = 0 ; -- commit label
        C_commit_rec constant  smallint : = 5000; -- commit rows each loop
        Cusor cur_ exp_consignment_order is
        Select  id,
                Language
                Swft_flag
         ---- 省略部分列
      From exp_consignment_order@tb_erp    -- dblink 连接源系统
      Where last_update_date >= Ld_start_date
        And last_update_date <= Ld_end_date
       --- Program Start
      Pkg_global.sp_exp_consignment_order(Lv_program_name)
      Pkg_global.sp_gb_program_date('OM', Ld_start_date, Ld_end_date);
      Execute immediare 'TRUNCATE TABLE STGDM.exp_consignment_order';
      For rec_exp_consignment_order in  cur_exp_consignment_order
      Loop
        Begin
          In_key1: = rec_exp_consignment_order.id
          In_Key2: = rec_exp_consignment_order.Language
           Insert intoexp_consignment_order
             (ID, Language,source_lang --- 省略部分列)
          Values
           (rec_exp_consignment_order.id, rec_exp_consignment_order.Language)
          Ln_ins_counter : = Ln_ins_counter + 1
          Ln_commit_label : = Ln_commit_label + 1
        End
        If Ln_commit_label >= C_commit_rec
        Begin
          Commit;
          Ln_commit_label : = 0;
        End if
    End loop
    Commit;
     -- Progarm Complete
    Pkg_global.sp_gb_progarm_complete(lv_program_name,ln_upd_counter,ln_ins_counter)
    Exception
    When others then
     --- program Error Raise
    Lv_key_information : = lv_table_name ||':'||lv_key_name1||'='to_char(ln_key1)||':'||lv_key_
    name2                        ||'='to_char(lv_key2);
    Pkg_global.sp_progarm_error_raise(lv_program_name,lv_key_information);
    End sp_ exp_consignment_order
```

加载临时存储区数据到 ODS 层可参照上面从源系统到临时存储区的加载过程，主要区别是如果需要处理 Purge/Delete，应增加 Purge_delete_Flag 字段。

2. 数据转换

数据从源系统加载到临时存储区，还需要经过一系列转换。Oracle 9i 内部数据转换主要采用三种方式：

- 使用 SQL 进行转换
- 使用 PL/SQL 进行转换

• 使用表函数进行转换

本例中数据转换主要采用 SQL，所以主要介绍 Oracle 9i 使用 SQL 执行合并的转换方法。在更新研发产品维表 Product_dev_dim 时，源系统具有和 Product_dev_dim 相同的结构，可通过以下代码实现。

```
Merge into  product_dev_dim pdd
Using product_dev pd
On(pdd.prod_id = pd.prod_id)
When matched then
Update set
Pdd.code = pd.code,
Pdd.name = pd.name
When not matched then
Insert
    (product_id,product_code,product_name,product_pre,effective,start_date,
    end_date)
Values
(pd.product_id,pd.product_code,pd.product_name,product_pre,pd.effective, pd.start_date,
pd.end_date)
```

3. 数据清洗

本例中数据清洗依照前面数据清洗的设计方案，即：

(1) 创建清洗脚本(脚本创建在 ODS 层)，实现代码如下：

```
CREATE TABLE CHK_RULE_DEF  (
   RULE_ID              NUMBER                          NOT NULL,
   RULE_REF             VARCHAR2(10),
   RULE_DESC            VARCHAR2(240),
   IS_ENABLED           CHAR(1),
   PROC_NAME            VARCHAR2(60),
   REMARK               VARCHAR2(240)
)
ALTER TABLE CHK_RULE_DEF
   ADD CONSTRAINT PK_CHK_RULE_MASTER PRIMARY KEY (RULE_ID)
/*==============================================================*/
/* Table: CHK_TABLE_RULE_DEF                                     */
/*==============================================================*/
CREATE TABLE CHK_TABLE_RULE_DEF  (
   TABLE_NAME           VARCHAR2(30)                    NOT NULL,
   MIN_NUM_ROW          NUMBER,
   MAX_NUM_ROW          NUMBER,
   PARAMETER1           VARCHAR2(100),
   PARAMETER2           VARCHAR2(100),
   PARAMETER3           VARCHAR2(100),
   PARAMETER4           VARCHAR2(100)
)
...
```

(2) 初始化清洗元数据,实现代码如下:

```
-- DELETE DATA
delete from CHK_TABLE_RULE_DEF;
delete from CHK_RULE_DEF;
delete from CHK_REFERENCE_KEY;
delete from CHK_FIELD_RULE_DEF;

-- INSERT META DATA FOR DATA CLEARING
insert into CHK_FIELD_RULE_DEF (TABLE_NAME, FIELD_NAME, FIELD_TYPE, NULLABLE, MIN_VALUE,
MAX_VALUE, CHK_STRING, BUSINESS_KEY, PARAMETER1, PARAMETER2, PARAMETER3, PARAMETER4)
values ('ACCOUNTS', 'REFERENCE', 'C', 'N', null, null, null, 'Y', null, null, null, null);
insert into CHK_FIELD_RULE_DEF (TABLE_NAME, FIELD_NAME, FIELD_TYPE, NULLABLE, MIN_VALUE,
MAX_VALUE, CHK_STRING, BUSINESS_KEY, PARAMETER1, PARAMETER2, PARAMETER3, PARAMETER4)
values ('ACCOUNT_CONTRACTS', 'REFERENCE', 'C', 'N', null, null, null, 'Y', null, null, null,
null);
insert into CHK_FIELD_RULE_DEF (TABLE_NAME, FIELD_NAME, FIELD_TYPE, NULLABLE, MIN_VALUE,
MAX_VALUE, CHK_STRING, BUSINESS_KEY, PARAMETER1, PARAMETER2, PARAMETER3, PARAMETER4)
values ('ADDRESSES', 'REFERENCE', 'C', 'N', null, null,null, 'Y', null, null, null, null);

insert into CHK_RULE_DEF (RULE_ID, RULE_REF, RULE_DESC, IS_ENABLED, PROC_NAME, REMARK)
values (1, '', 'Check proper surrogate key(check business key)', 'N', 'PKG_SANITY_CHECK.SP_
CHK_BUSINESS_KEY', null);
insert into CHK_RULE_MASTER (RULE_ID, RULE_REF, RULE_DESC, IS_ENABLED, PROC_NAME, REMARK)
values (2, '', 'Check unique of record', 'N', 'PKG_SANITY_CHECK.SP_CHK_UNIQUENESS', null);
insert into CHK_RULE_MASTER (RULE_ID, RULE_REF, RULE_DESC, IS_ENABLED, PROC_NAME, REMARK)
...
```

(3) 执行数据清洗程序,实现代码如下:

```
CREATE OR REPLACE PACKAGE PKG_DATA_CHECK AS

    PROCEDURE SP_CHK_CURRENT_IND;
    PROCEDURE SP_CHK_VALID_DATE;
    PROCEDURE SP_CHK_SURROGATE_KEY;
    PROCEDURE SP_CHK_BUSINESS_KEY;
    PROCEDURE SP_CHK_UNIQUENESS;
    PROCEDURE SP_CHK_FOREIGN_KEY;
    PROCEDURE SP_CHK_SKELETON_RECORD;
    PROCEDURE SP_ERRM_LOG(P_CHK_ID NUMBER,P_RULE_ID NUMBER,P_TABLE_NAME VARCHAR2, P_CHK_
SQL VARCHAR2,P_SQL_ERRM VARCHAR2) ;
    PROCEDURE SP_CHK;  -- desc: the check loop
    FUNCTION F_GET_BUSINESS_KEY(P_TABLE_NAME VARCHAR2)  RETURN VARCHAR2;
    PKG_SDATE VARCHAR2(10) := TO_CHAR(SYSDATE,'dd/mm/yyyy');  /* today string */
    TYPE REF_CUR  IS REF CURSOR;                              /* Ref Current Type */
    REPORT_APP_ID CONSTANT VARCHAR2(20):= 'GSM';              /* application name */
    DATA_SCHEMA CONSTANT VARCHAR2(30):= 'DATAPLAT';           /* the data schema */
```

```
        SP_SCHEMA CONSTANT VARCHAR2(30) : = 'DP_RECON'; /* the sanity check schema(not used) */
        PKGB_CHECK_ID NUMBER;                              /* current check id */
        PKGB_RULE_ID NUMBER;                               /* current rule id */
        DEBUGING BOOLEAN  : =  FALSE;  /* debug mode   True: when error accured will output
    error message
                                  False: when error accured not output the error message */
      END PKG_DATA_CHECK;
      CREATE OR REPLACE PACKAGE BODY pkg_data_check
      AS
          v_sql                          LONG;
        FUNCTION f_get_business_key (p_table_name VARCHAR2)
          RETURN VARCHAR2
        IS
          v_return                        VARCHAR2 (1024);

          CURSOR cur_business_key (p_table VARCHAR2)
          IS
            SELECT field_name
              FROM chk_field_rule_def
             WHERE UPPER (business_key)  =  'Y'
               AND UPPER (table_name)  =  UPPER (p_table);
          -- v_business_key                 VARCHAR2 (30);

          BEGIN
            v_return : =  '';
            OPEN cur_business_key (p_table_name);
            LOOP
                FETCH cur_business_key
                 INTO v_business_key;
                EXIT WHEN cur_business_key % NOTFOUND;
                IF v_return !=  '' THEN
                  v_return : =  v_return || '||'','''||@TABLE.'|| v_business_key;
                ELSE
                  v_return : =  '@TABLE.'|| v_business_key;
                END IF;
            END LOOP;
            CLOSE cur_business_key;
            RETURN v_return;
         EXCEPTION
           WHEN OTHERS THEN
             IF cur_business_key % ISOPEN THEN
               CLOSE cur_business_key;
             END IF;
             RETURN '';
         END;
     …
```

4. 数据装载

本例中，ETL 按照 Staging → ODS → Baseline → Data Mart 的流程，所有程序由 Scheduler 统一调度。以质量绩效分析为例，其装载过程包括初始装载（Initial Loading）和

增量装载(Incremental Loading)两部分。初始装载相对简单,这里主要介绍增量装载,如图 3.18 所示为增量装载的完整流程。

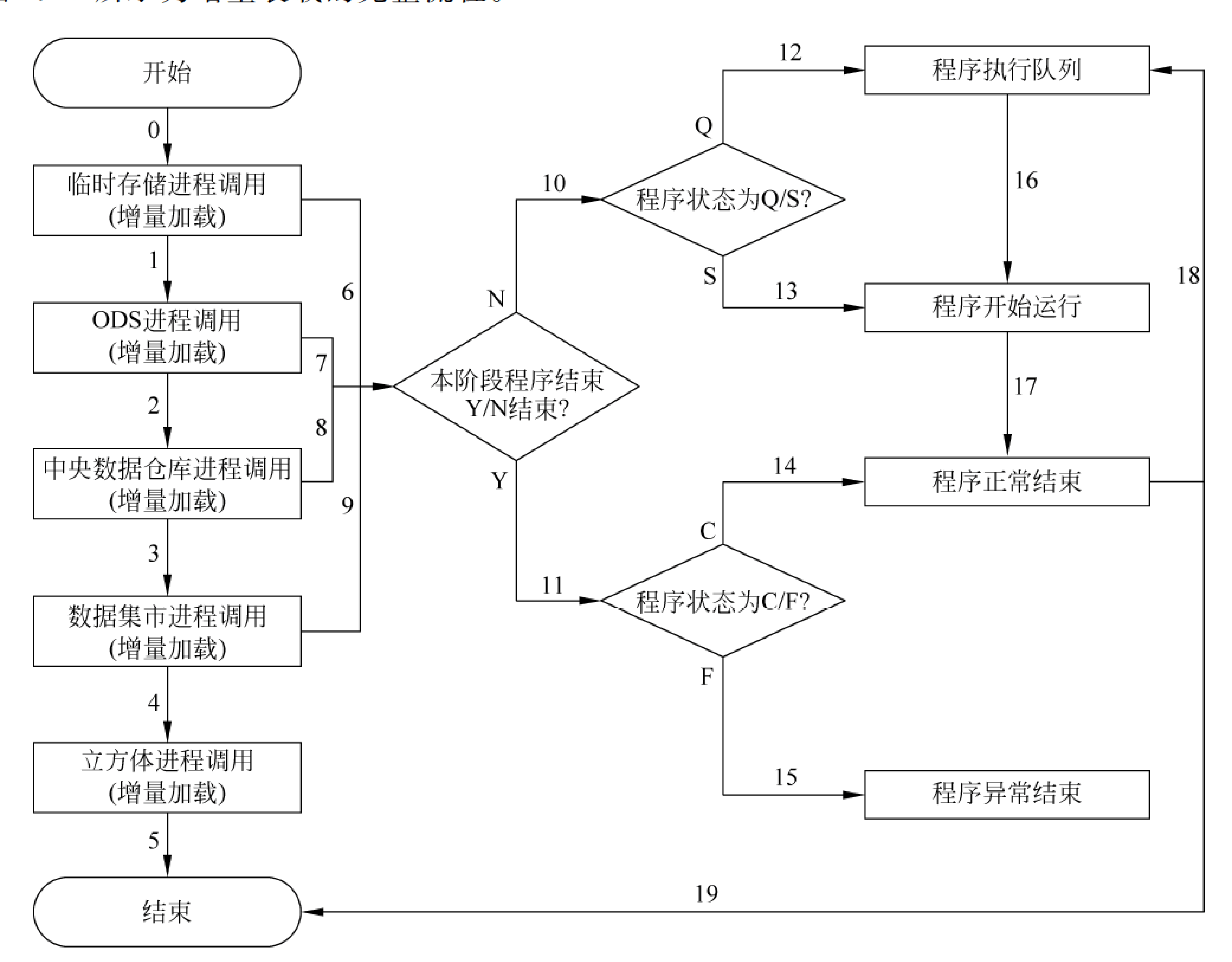


图 3.18 增量装载流程

其中:

- 步骤 1~5 是增量调度程序总调度的 5 个阶段。
- 步骤 6、7、8、9 增量调度运行每个阶段的运行状态有 4 种(Q/S/C/F),其中 Q(Queued)表示程序进入执行队列中,S(Started)表示程序开始运行,C(Completed)表示程序正常结束,F(Failed)表示程序异常结束。
- 程序运行的 4 种状态保存在表 program_reference_table 中,调度程序判断表 program_reference_table 中各程序的状态。
- 步骤 16:增量调度程序被 Server 调用,进入队列中,在 program_reference_table 表中本程序状态为 Q。
- 步骤 17:增量程序进入执行队列后被执行,执行时 program_reference_table 表中本程序状态为 S。
- 步骤 18:增量程序执行完毕后,在 program_reference_table 表中本程序状态为 C,同时进入下一个程序的执行。
- 步骤 19:当所有的程序执行完毕后,增量装载结束。

第 4 章　数据仓库应用——OLAP 和 OLAM

由数据仓库的定义可知数据仓库是面向决策支持的，即各种分析型应用，如联机分析处理、数据挖掘、商业智能(Business Intelligence，BI)和客户关系管理等，数据仓库的数据通过 OLAP 和 DM 后，转换为信息，并最终形成知识，为科学决策提供辅助支持。

4.1　OLAP

1. 定义

OLAP 是基于数据仓库的一种数据分析和处理技术，也可以看成是基于数据仓库的一种软件工具。OLAP 侧重于对决策者和高层管理人员的支持，可以根据分析人员的要求，快速、灵活地实现大量数据的复杂查询，并以一种简单、直观的形式展现查询结果。基于数据仓库实施 OLAP，可以帮助企业管理者掌握企业经营状况，了解市场需求，制定科学决策，提高核心竞争力。OLAP 的目标是满足决策支持或满足在多维环境下特定的查询和报表需求。

OLAP 最早是由关系数据库之父 E. F. Codd 于 1993 年提出的。当时，Codd 认为 OLTP 已不能满足终端客户对数据库查询分析的需要，SQL 对大型数据库进行的简单查询也不能满足客户分析的需求。客户的决策分析需要对关系数据库进行大量计算才能完成，而查询的结果并不能满足决策者提出的需求。因此 Codd 提出多维数据库和多维分析的概念，即 OLAP。

2. 分类

根据数据存储方式的不同，OLAP 可分为 ROLAP(Relational OLAP)、MOLAP(Multi-Dimensional OLAP)和 HOLAP(Hybrid OLAP)三类。ROLAP 是指基于关系数据库的 OLAP，以关系数据库为核心，以关系型结构进行多维数据的表示和存储。ROLAP 将多维数据库的多维结构划分为两类表：一类是事实表，存储数据和维关键字；另一类是维表，即对每个维至少使用一个表来存放维的层次、成员类别等维的描述信息。维表和事实表通过主关键字和外关键字关联起来，形成了“星型模式”。对于层次复杂的维，为避免冗余数据占用过大的存储空间，可以使用多个表描述，此时星型模式扩展为“雪花型模式”。MOLAP 是指基于多维数据组织的 OLAP，以多维数据组织方式为核心，即 MOLAP 使用多维数组存储数据。多维数据在存储中形成立方体结构，在 MOLAP 中对立方体的旋转、切块和切片是产生多维数据报表的主要技术。HOLAP 是指基于混合数据组织的 OLAP，如低层是关系型的，高层是多维的。这种方式具有更好的灵活性。实际上，HOLAP 是 MOLAP 和 ROLAP 的折中。对于常用的维和维层次，在 HOLAP 中使用多维数据表记录，对于用户不常用的维和数据，HOLAP 采用类似于 ROLAP 的星型模式存储。当用户需要访问不常用的数据时，HOLAP 就会把简化的多维数据表和星型模式相拼合，从而得到完整的多维数据表。HOLAP 多维数据表中的维度少于 MOLAP 多维数据表的维度，数据存储容量也小于

MOLAP 方式。但是,在数据存取速度上 HOLAP 又低于 MOLAP。HOLAP 的主要性能都介于 MOLAP 和 ROLAP 之间,其技术复杂度高于 ROLAP 和 MOLAP。

根据数据组织方式的不同,OLAP 可分为基于多维数据库的 OLAP(MD OLAP)和基于关系数据库的 OLAP(ROLAP)两种。前者响应速度快、执行效率高,但由于结构的局限,灵活性不够。与之相比,后者建立在大量现有数据库(数据仓库)的基础上,灵活性、扩展性要高得多,并且支持大数据量和高维的能力也要强于前者。因此,虽然在响应速度、执行效率上差一点,仍然得到广泛应用,现有的 OLAP 工具大多基于后者。

目前,针对 OLAP 的研究相当活跃,对 OLAP 的理解也不断深入。有人提出了 OLAP 更为简洁的定义,如 Nigel Pendse 提出的 FASMI(Fast Analysis Of Shared Multidimensional Information),所采用的技术包括客户端/服务器结构、时间序列分析模型、并行处理、面向对象、数据存储和多线程技术等。

3. 典型操作

OLAP 对数据仓库数据的操作基于多维视图(或称超立方体)。对立方体的典型操作包括切片、切块和旋转等,如图 4.1 所示。

- 切片是指选定多维数组的一个二维子集。
- 切块是指选定多维数组的一个三维子集。
- 旋转是指改变一个立方体显示的维方向,使人们可以从不同的角度更加清晰、直观地观察数据。

此外还包括其他一些操作,如:

- 上卷是指沿某一个维的概念分层向上归约,并且通过维规约,在立方体上进行聚集。
- 下钻是上卷的逆操作,由不太详细的数据到更详细的数据。
- 钻取是指对多个事实表进行查询。
- 钻透是指对立方体操作时,利用数据库关系,钻透立方体的底层,进入后端的关系表。

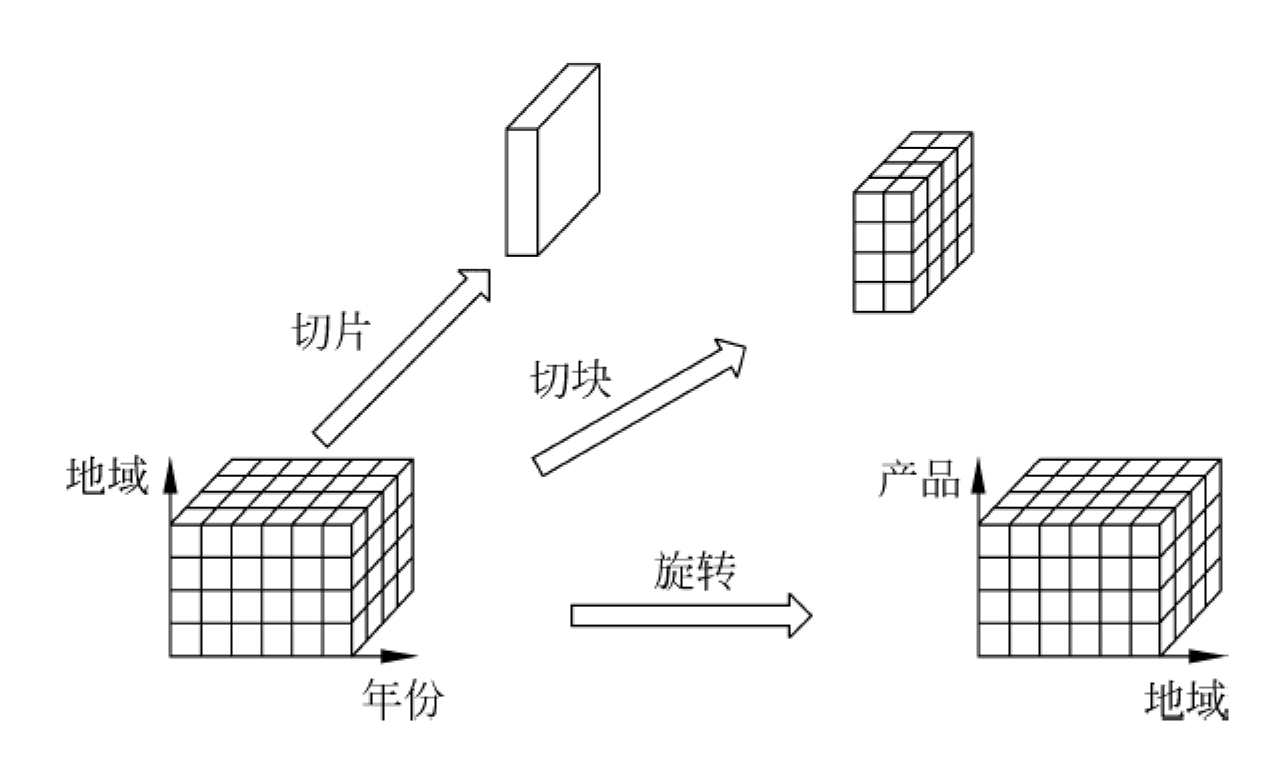


图 4.1　立方体的典型操作

4. 主要特点

OLAP 是数据仓库的分析展示工具,建立在数据多维视图的基础上。其主要特点一是在线(Online),体现为对用户请求的快速响应和交互式操作;二是多维分析,这也是 OLAP 技术的核心所在。

OLAP 主要通过多维方式实现数据分析、查询和报表，不同于传统的 OLTP。首先，OLTP 主要面向公司员工，OLAP 则主要面向公司领导层；其次，OLTP 应用主要完成客户的事务处理，其数据基础是操作型数据库，如民航订票系统、银行储蓄系统等，通常需要进行大量的更新操作，同时对响应时间要求较高。而 OLAP 是以数据仓库或数据多维视图为基础的数据分析处理，是针对特定问题的联机数据访问和分析，一般不进行数据修改只是查询，其应用主要是分析用户的当前及历史数据以辅助决策，典型的应用包括对银行信用卡风险的分析与预测、公司市场营销策略的制定等，主要是进行大量的查询操作，对响应时间的要求不太严格。

5．实现途径

OLAP 实现通常采用三层客户端/服务器(Client/Server，C/S)结构。第一层是数据仓库服务器，实现与基层运营的数据库系统的连接，完成数据的一致性和共享；第二层是 OLAP 服务器，根据最终服务请求分解成 OLAP 的各种操作，并使用数据仓库中的数据完成这些操作；第三层是前端展现工具，用于将 OLAP 服务器处理得到的结果以直观的方式，如多维报表、饼图和三维图等展现给最终用户。

6．评价标准

1993 年，E. F. Codd 提出了关于 OLAP 的 12 条评价标准，旨在加深对 OLAP 的理解，后来扩充到 18 条。对于设计数据仓库以及使用 OLAP 的用户而言，理解这些评价标准是十分必要的。

18 条准则的具体内容如下：

准则 1　多维性，能对多维数据进行切片、切块和旋转等操作；

准则 2　直观性，能为用户提供直观、易用的数据操作；

准则 3　可访问性，OLAP 以合适的方式存储数据，便于用户访问；

准则 4　解释性批处理，在 OLAP 中常由 OLAP 引擎或服务器上存储立方体的混合工具实现；

准则 5　OLAP 分析模型，在高层获取 OLAP 所支持的分析数据，包括静态描述性报告、解释性分析、假设性分析和预测性分析等；

准则 6　客户端/服务器结构性，用户通过客户端与服务器的松散耦合实现 OLAP；

准则 7　透明性或开放性，OLAP 及其数据源对用户透明和开放；

准则 8　多用户性，要求 OLAP 在实际应用中保证数据的完整性和安全性，并能够进行数据的并发处理；

准则 9　处理非正规数据，要求系统满足“强聚合，弱耦合”的标准；

准则 10　存储 OLAP 结果，要求将决策分析和数据源分开；

准则 11　提取丢失值，是系统处理空值的一种方式；

准则 12　处理丢失值，要求 OLAP 引擎忽略已经提取的丢失值；

准则 13　弹性报告，要充分反映数据的多维特征，具有较强的灵活性；

准则 14　一致性报告，即要求 OLAP 能够为用户提供时间可预计的报告；

准则 15　对物理层的自动调整，其为关系模型的标准；

准则 16　通用维，即在结构和操作能力方面完全一致的维；

准则 17　无限维与聚合层，OLAP 维数不应该小于 15，且在任意路径建立任意多个聚

合层次,给定路径的概括级数据也是有限的;

准则 18 无限制跨维操作,要求能够在维之间进行符号操作,而不仅仅是对可测量数据的操作。

上述评价标准仍存在较大争议,例如第 16 条除了时间维,多数维都有自己的特性;而第 17 条准则在现实中很难实现,否则会导致数据在有限空间内急剧膨胀。

7. 主流工具

目前,市场上主流的 OLAP 及前端展现产品几乎都是在 1993 年之前出现的,有的甚至已有三十多年的历史,如 Cognos(PowerCenter)、Hyperion(Essbase)、微软(Analysis Service)、Business Object 以及 MicroStrategy 等几大厂商的产品。

综合业界多方面观点,在此将对 Cognos、Hyperion 和 Business Object 这三家厂商的产品在整体解决方案、OLAP 工具、系统稳定性以及集成等方面进行综合比较。

1) Cognos 8

Cognos 8 解决方案整合在一个产品中,基于一种 Web 架构上,面向企业所有数据源,面向所有用户提供完整的信息处理与展现功能,包括了即席分析功能、查询功能、报表功能、仪表盘功能、事件管理功能以及 BI 管理功能等。Cognos 8 采用 SOA 架构,统一了 Web 应用架构和元数据,能够访问企业的所有数据源,为企业所有用户提供了基于纯浏览器的、全面的 BI 功能。

- 前端产品:主要包括 Report Studio,Analysis Studio 以及 Query Studio 等。
- OLAP 工具:Cognos Powerplay 以桌面 OLAP 开始,一直以轻便、快捷的操作闻名,虽然 Poweplay 早已演变成 C/S 结构的 OLAP 服务器,但其轻便的特点还是延续下来,而且提供可以简洁部署且具有交互性的 Framework Manager 界面。

其优缺点如下:

- 优点——易于集成、部署和使用,经过简单培训后就能进行设计与开发。
- 缺点——目前国内没有分公司,只有国家信息中心下属的优信佳公司独家代理,其服务能力有限。

2) Hyperion System 9

Hyperion System 9 BI+实现 BI 标准化策略的方法是用统一系统满足多种多样的报表和分析需求,即企业内每个部门的每位用户,都能通过易于操作和维护的单一系统按需生成报表或进行分析。这样,企业内的信息使用者和生产者就获得了快速生成、访问和共享重要信息的便利,增强了工作的灵活性,而他们也需要这些重要信息来更快、更好地做出业务决策。简单而功能强大的用户界面使用户在自行创建报表时对 IT 支持和依赖降到了最低。产品模块化的架构极大地降低了用户对 IT 支持的需求,通过 Hyperion System 9 BI+平台,用户只需在服务器端部署和管理即可。

- 前端产品:主要包括 BI+ Web Analysis 和 BI+ Interactive Reporting。
- OLAP 工具:Essbase 作为老牌的 OLAP 服务器,是一个比较复杂的产品。所谓复杂有两层含义,一是提供丰富的 API,允许充分定制开发;二是开发的难度较大,不易部署。虽然其产品性能很高,但即使厂商的技术人员也很难掌握其烦琐的细节技术。

其优缺点如下:

• 优点——BI+ Web Analysis 查询灵活,不需编程,适合业务人员使用。BI+ Interactive Reporting 适合复杂查询和报表设计。结合 Essbase,性能较高。
• 缺点——安装配置非常麻烦,使用 BI+ Interactive Reporting 必须会 JSP 语言,操作性较差,各种图形饱和度差些。

3) Business Object(简称 BO)

BO 是集查询、报表和 OLAP 展现为一身的智能决策支持工具,它使用独特的语义层和动态微立方等技术表示数据库中的多维数据,具有较好的查询和报表功能,提供钻取等多维分析技术,支持多种数据库,同时还支持基于 Web 浏览器的查询、报表和分析决策。

• 前端产品:主要包括 Web Intelligence 和 Crystal Reporting(水晶报表),还有仪表板和绩效管理工具。
• OLAP 工具:虽然 BO 在不断增加新的功能,但严格地讲,只能算是一个前端工具。也许正因为如此,几乎所有的数据仓库解决方案都把 BO 作为可选的数据展现工具。

其优缺点如下:

• 优点——用户界面美观,图形饱和度高,拥有多种新颖仪表板和记分卡。
• 缺点——其服务要求不能出现网络中断,否则需要重启。程序安装后不能删除,否则会造成系统崩溃,仪表板之间的集成度稍差些。报表和分析工具较复杂,用户很难自己设计所需的报表,开发与维护成本较高。

4.2 OLAM

数据挖掘与 OLAP 不同,主要体现在其分析数据的深入和分析过程的自动化,自动化的含义是指分析过程不需要用户参与。这是一把双刃剑,在实际中,用户也希望参与到挖掘中,如仅对数据的某一子集进行挖掘,以及对不同抽取、集成水平的数据进行挖掘,还有想根据自身的需要动态选择挖掘算法等。可见,OLAP 与数据挖掘各有所长,如果能将二者结合衍生出一种为数据挖掘服务的具有新型 OLAP 功能的应用,将更能适应实际的需求,联机分析挖掘(OnLine Analytical Mining,OLAM)正是这种结合的产物。

1997 年 Han J. W. 提出了 OLAM 的概念,他把 OLAM 定义为 OLAP Mining,其含义是将 OLAP 和数据挖掘技术结合起来,在多维数据模型即数据立方体的基础上对外提供数据分析和知识发现应用,即在 OLAP 基础上,对数据分析算法进行扩充,把数据挖掘算法引入到多维数据模型的数据环境中,并把这种思想在其研制的 DB Miner 系统中加以实现。

Han J. W. 提出的 OLAM 的研究方向是数据分析算法和数据挖掘算法如何与数据立方体高效地结合,解决多维数据环境的数据挖掘。但是对于如何在系统体系结构上把 OLAP 和数据挖掘有机地结合起来,即在异构、海量的环境中快速响应用户的数据分析和数据挖掘请求的问题没有做深入研究。

就 OLAP 和数据挖掘技术结合的系统集成度而言,可以分为松散的集成和紧密的集成两种,即:

(1) 松散的集成是指系统的集成度不高,即把分立的 OLAP 系统和数据挖掘系统组合在一起,两者没有统一的逻辑模型和任务/事务模型。

(2) 紧密的集成是指系统采用统一的逻辑模型、任务/事务模型、数据定义语言和数据操作原语，对于数据密集和耗时较多的操作系统进行统一的调度和优化，从而在系统内核上将 OLAP 和数据挖掘有机结合在一起。

Han J. W. 提出的 OLAM 以及其实现的 DB Miner 系统是一种松散的集成。其实，早在 OLAM 概念之前，在数据挖掘领域已有将 OLAP 与数据挖掘结合起来提供更优质的数据分析和决策支持工具的思路。有许多 OLAP 产品在功能上添加了数据挖掘能力，在具体实现方式上可以分为两类：

(1) 在现有 OLAP 产品基础上，通过系统的改造增加数据挖掘功能，如 Business Object 产品中的决策树分析、DB Miner 系统中的数据挖掘算法工具箱。在这种实现方式中 OLAP 与数据挖掘的结合松散，拼装的痕迹明显，对系统整体体系结构的考虑较少，不能同时充分发挥两者的优势。

(2) 把数据挖掘算法集成在系统的底层功能中，OLAP 与数据挖掘结合紧密，Microsoft 公司的 SQL Server 2000 中的关联分析方法在数据库端的集成就是紧密集成这种方式的初步尝试。

推动 OLAM 发展的原始驱动力主要体现在如下几个方面：

(1) 分析需要的数据是一些经过净化、集成处理的数据，通常这种处理过程也是昂贵的。而数据仓库作为 OLAP 的数据源，存储的就是这样的数据。它能为 OLAP 提供数据，当然也可以为数据挖掘提供数据。

(2) 数据仓库是一项崭新的技术，很多人在研究它，围绕着它有许多工具或是体系结构。而数据挖掘作为数据分析工具的一种，不是孤立的，必然与其他工具发生联系。因此，考虑到如何最大限度地利用这些现成的工具，也是 OLAM 发展之初所关心的问题。

(3) 成功的数据挖掘需要对数据进行钻探性分析。例如，挖掘所需的数据可能只是一部分、一定范围的数据。因此，对多维数据模型的切片、切块和下钻等操作，同样可以应用于数据挖掘中。换言之，可以将数据挖掘建立在多维模型(或称超立方体)基础上。

(4) 用户的参与对数据挖掘的重要性。动态地提出挖掘要求、选择挖掘算法，因此可以将 OLAP 的 C/S 结构应用于数据挖掘。

4.2.1 体系结构

OLAM 的挖掘分析是建立在数据仓库基础上的，数据挖掘所需要的数据在挖掘进行之前就已经过预处理并存放在数据仓库中，这在很大程度上提高了数据挖掘的响应速度。

此外，具体的数据挖掘算法与数据源之间的关系完全透明。用户在进行挖掘操作时，首先根据算法要求从数据仓库中提取相关数据，然后进行运算，挖掘产生的结果即学习得到的知识，则写回到数据仓库的相应库表中，以立方体的形式展现给用户，如图 4.2 所示。

这里有两个问题需要详细讨论，即：

一是算法选择问题。如前所述，OLAM 发展的源动力之一是给予用户一定算法选择灵活性。因为对于同一个功能一般可以采用多种算法实现，例如分类可以是基于统计学，或基于机器学习或基于神经网络。不同的算法针对不同的问题空间有不同的性能，如何选择最

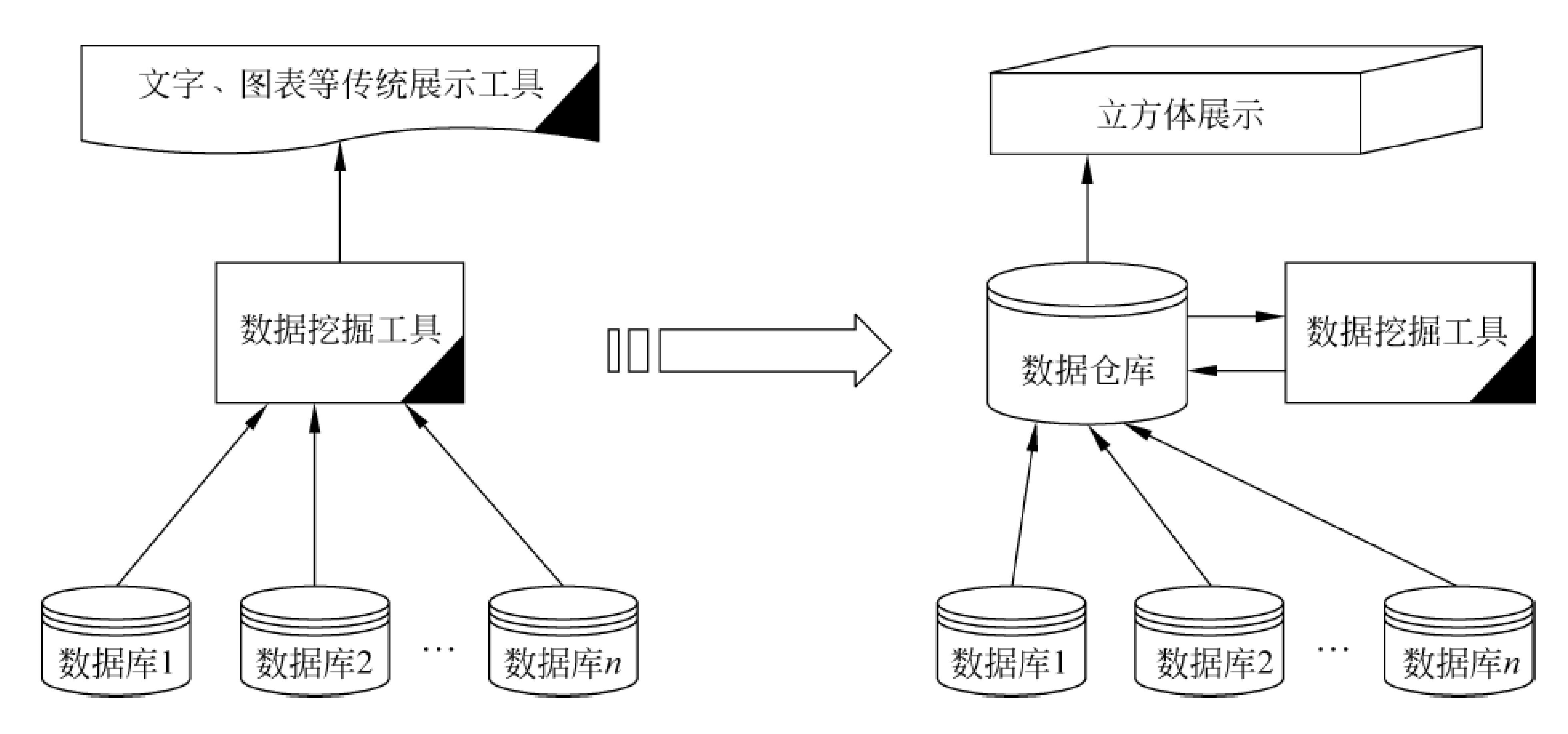


图 4.2　OLAM 结构演变示意图

高性能的算法,是用户需要考虑的问题。另一方面,OLAM 的最终用户往往对具体算法并不了解,因此提供太大的灵活度反而让用户束手无策。实际的做法是针对 OLAM 具体的问题空间,在充分实验的基础上,仅提供有限的几种算法参与知识发现过程,并将结果反馈给用户,让用户选择最贴近实际的结果。这在一定程度上避免了用户与挖掘算法的直接接触。

二是源数据钻探问题。我们知道,传统的数据挖掘建立在各种数据源之上,在挖掘之前,往往需要花费大量的时间进行数据预处理,这大大降低了系统的响应性能。而 OLAM 建立在数据仓库之上,挖掘所需的源数据事先已存放在数据仓库中。然而,有些挖掘算法往往需要底层的详细数据,如分析电信客户行为时需要客户的通话详单,这些数据一般数据量巨大且更新频繁,不可能将其全部复制到数据仓库中。一是因为系统容量有限,二是由于系统不堪频繁数据更新的重负。实际上,还是将这样的数据存放在各个事务数据库中,如前所述数据仓库在物理上也可能就是原来的事务数据库。因此,这些在实际中根据需要存储分析或挖掘所需数据的事务数据库,在逻辑上构成了一个虚拟数据仓库,如图 4.3 所示。

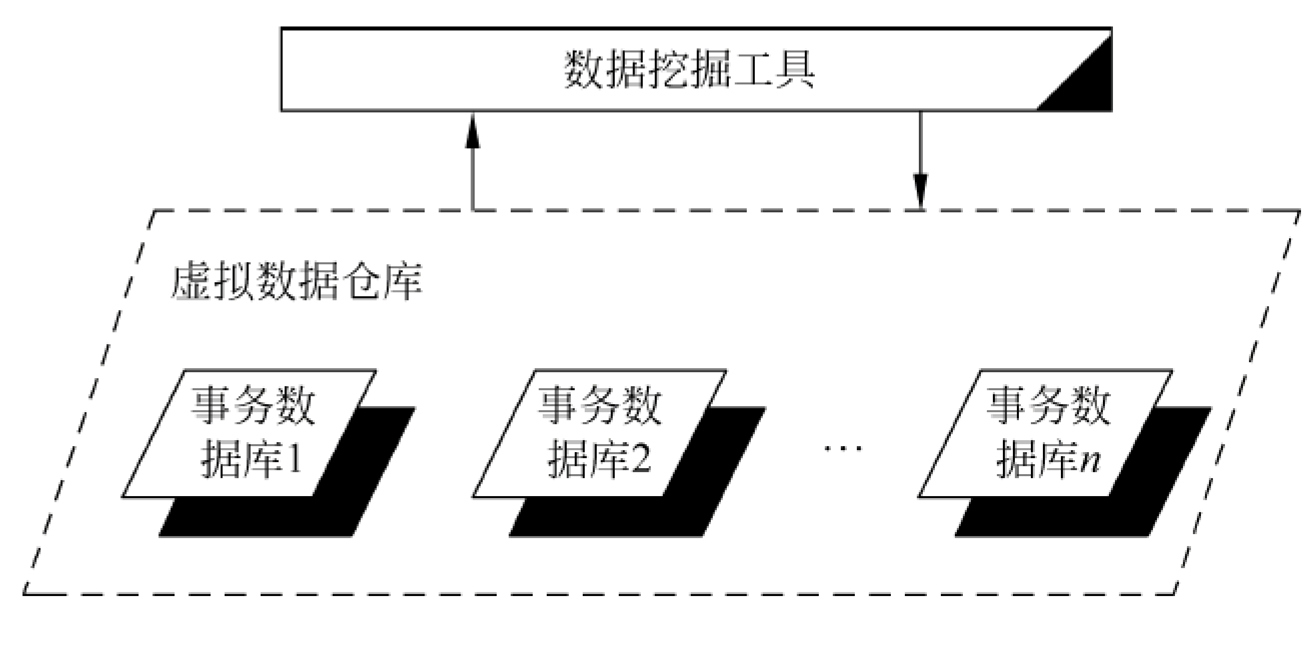


图 4.3　虚拟数据仓库示意图

OLAM 的挖掘分析是建立在超立方体基础上,而且在实际应用中,尽管 OLAM 的多维计算可能需要更多的维度和更强大的访问工具,但是 OLAP 的立方体和 OLAM 的立方体之间并没有本质的区别。

4.2.2 特点

建立在庞大数据库或数据仓库基础上的 OLAM 在实现过程中面临的最大挑战是执行效率的提高和对用户请求的快速响应。目前还没有 OLAM 的产品出现,对 OLAM 应具备的特点也众说纷纭。但是,针对 OLAM 的基本结构以下几点是必要的。

(1) OLAM 建立在多维数据库和 OLAP 的基础上,因此应能方便地对任何数据或不同抽象级别的数据进行挖掘。这是借助 OLAP 对超立方体进行切片、切块和下钻等操作实现的。另外,如果需要 OLAM 还可以直接访问存储在底层数据库中的数据。总之,借助于 OLAP 的支持,OLAM 能对任何数据进行挖掘。

(2) 用户对挖掘算法具有动态选择的权力,在传统的关系数据库中,对同一个主题,任何不同的查询过程所得到的结果是相同的。而数据挖掘则不然,对同一个问题,运用不同的挖掘算法,得出的结果可能大相径庭。因此有必要给予用户以动态选择挖掘算法的权力。此外,有些用户针对自己的问题,可能有一套独特的挖掘算法,并希望嵌入到 OLAM 中。因此,OLAM 应该具有通用接口,便于与其他工具或算法相衔接。

(3) OLAM 建立在多维数据视图的基础上,因此基于超立方体的挖掘算法是其核心。超立方体计算与传统挖掘算法的结合使得数据挖掘有了极大的灵活性和交互性。这里所说的超立方体计算一般是指切片、切块、下钻和旋转等操作。而传统挖掘算法是指关联、分类、聚类和预测等挖掘算法。根据超立方体计算和数据挖掘所进行的次序的不同组合可以有以下几种模式,即:

- 先进行立方体计算,后进行数据挖掘。在进行数据挖掘前,先对多维数据进行一定的立方体计算,以选择合适的数据范围和恰当的抽象级别。
- 先对多维数据进行数据挖掘,然后再利用立方体计算对挖掘结果做进一步的深入分析。
- 立方体计算与数据挖掘同时进行,在挖掘的过程中可以根据需要对数据视图进行相应的多维操作,这意味着同一个挖掘算法可以应用于多维数据视图的不同部分。

(4) 回溯和书签功能。OLAM 的挖掘过程是对多维数据视图的一个不断深入的过程。实际中,用户很有可能因为算法的复杂而在超立方体中“迷失方向”。因此 OLAM 的挖掘算法应能给用户提供退回上次操作初始状态的回溯及书签功能。

(5) 与 OLAP 类似,OLAM 也采用 C/S 体系结构,这使得其具有较高的执行效率和较快的响应速度。但由于一般挖掘算法都复杂且耗时,因此在执行效率和挖掘准确性两者之间需要权衡。一般情况下,OLAM 与用户频繁且高速地交互,应选择合适的挖掘算法和数据空间,一旦这两点确定了,在特定数据空间上再执行用户选定的复杂的挖掘算法,以保证最终结果的正确性。

此外,OLAM 还应具有灵活的可视化工具和良好的扩展性。OLAM 的结构复杂,在实际应用中要与多个模块或工具交互。例如,OLAM 可能与一个统计软件包相结合,或者系统本身功能扩展使之适合于地理数据、文本数据或者商业数据的挖掘。因此,OLAM 接口的标准化和通用性则显得尤为重要。OLAM 是 OLAP 与数据挖掘相结合的产物,它兼具 OLAP 多维分析的在线性、灵活性和数据挖掘对数据处理的深入性,是具有数据挖掘功能

的数据仓库,也是数据仓库应用工具未来发展的方向。

4.2.3 基于 Web 的 OLAM

基于 C/S 架构的 OLAP、OLAM 的一个重要特点是与用户的交互性。WWW 也是基于这样的模式。此外,由于用户前端展示工具的一致性(各种浏览器),使其具有更大的开放性。基于 Web 的 OLAM 是 Web 技术与 OLAP、OLAM 的结合,也是 Web 数据库下一步发展的目标。

可以把基于 Web 的 OLAM 看作是能提供多维分析和挖掘功能的 Web 数据库应用。OLAM 以应用程序服务器的形式安装于网络,浏览器端的用户可以通过 Web 服务器访问 OLAM 服务器。

如图 4.4 所示,WWW 服务器是实现 OLAM 功能的中枢,其执行流程的大概步骤如下:

(1) 浏览器端用户通过 HTML 文件中的表单提出数据分析挖掘请求并传递给 WWW 服务器。

(2) WWW 服务器端调用相应的应用程序如 CGI、ISAPI、NSAPI 等,并根据需要激活 OLAM 服务程序。

(3) OLAM 服务器引擎将立方体操作转换为 SQL 请求,并交付 DBMS/DWMS 执行。

(4) WWW 服务器将结果反馈给用户。

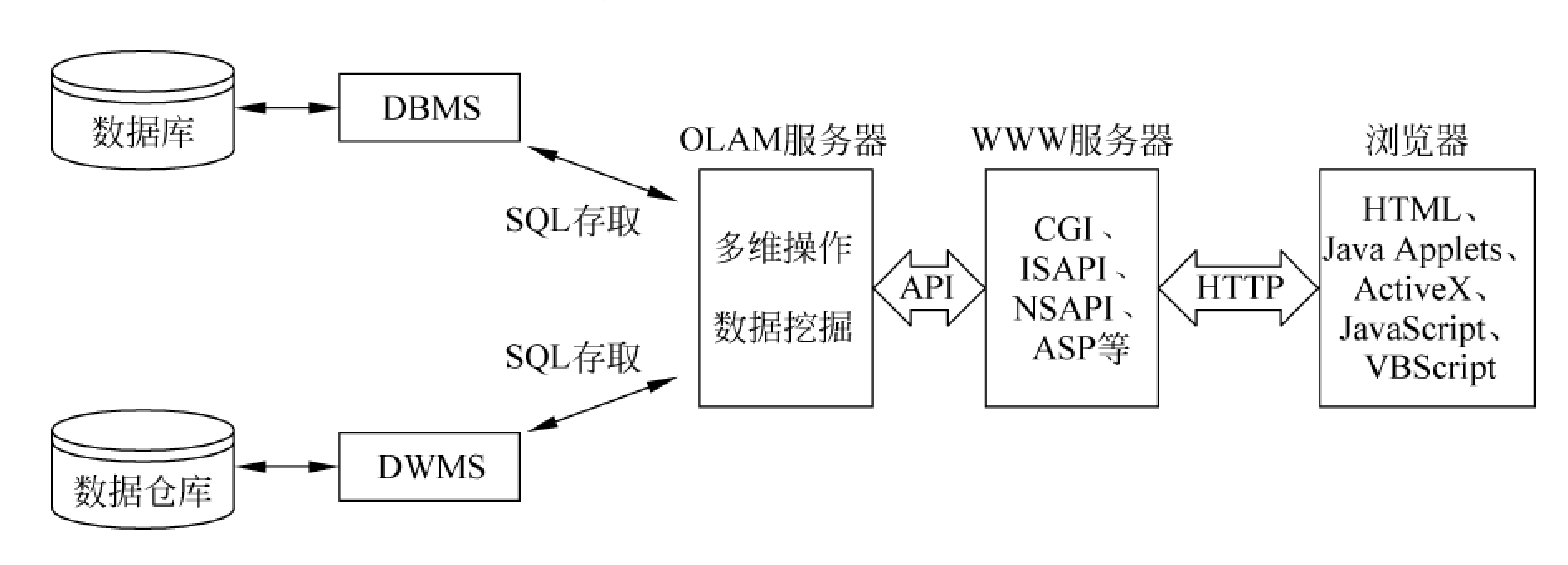


图 4.4 基于 Web 的 OLAM 结构框图

实现交互式的数据库应用系统,在服务器端一般是执行相应的 CGI 或 ISAPI 程序,但开发 CGI 和 ISAPI 复杂而低效,并且对程序员要求也较高。目前较为先进的是采用 ASP (Active Server Pages)技术。另外,考虑到最大限度地减少通信量及增加灵活性,在浏览器端引入了 Java Applets、ActiveX 或 VBScript 等组件。

目前,基于 Web 的 OLAM 应用还处于起步阶段,这一领域还有很多问题有待研究。例如:

(1) Web 数据以多种形式存在,一般是超文本文件,而且结构性较差。迄今为止,Internet 的信息描述语言已由最初的 SGML、HTML 发展到现在的 XML 和 DHTML。但是,由于浏览器的不同,很难在各种语言的解释执行方式上取得一致。因此,数据描述语言的标准化显得尤为重要。

(2) 基于 Web 的 OLAM 前端展示工具。浏览器虽然具有界面统一、易于操作等特点,但仅限于提供交互式操作,很难构造复杂应用。因此,能否开发一种接近人类自然语言的 Web 查询语言,成为研究热点之一。

(3) 系统执行效率和响应速度是用户最关心的问题,也是基于 Web 的 OLAM 在实际中遇到的最大挑战。影响这一性能的主要是两大因素,一是物理网络的传输速度;二是服务器端的分析挖掘算法的执行效率。随着网络技术的提高,前者已不足为虑,而后者因为大部分多维分析算法,如对超立方体的切片、切块、旋转等操作以及各种数据挖掘算法都是复杂而耗时的,因此对 OLAM 服务器提出了很高、有时甚至是很难满足的要求。

此外,还有一些非技术因素也是基于 Web 的 OLAM 在今后的发展中需要认真解决的问题,如信息安全性、保密性以及版权和收费问题等。

除了 OLAP 和 OLAM 之外,数据挖掘也是数据仓库的一个重要应用。下面将详细介绍数据挖掘。

第二篇

数据挖掘

第5章　数据挖掘基础
第6章　聚类分析
第7章　分类和预测
第8章　关联分析
第9章　Web挖掘
第10章　数据挖掘实例

第5章 数据挖掘基础

20世纪末，随着Internet的普及，全球信息量以惊人的速度急剧增长，据估计每18个月就增加一倍。许多组织机构的IT系统都存储了大量的数据。目前的数据(仓)库系统虽然可以高效地实现数据的录入、查询和统计等功能，但却无法发现海量数据中隐藏的关系和规则，无法预测未来的发展趋势。为了充分利用资源，从海量数据中发现隐藏的知识和规律，数据挖掘应运而生并显示出极其强大的生命力。

数据挖掘作为一门新兴的交叉学科，涉及数据库、数据仓库、统计学、机器学习、可视化、信息检索和高性能计算等诸多领域，其他相关的领域还包括人工智能(Artificial Intelligence，AI)、模式识别、空间数据分析、图像处理、概率论、图论和逻辑推理等。

5.1 概述

数据挖掘的提出是在20世纪80年代，它是一个新兴的、面向商业应用的AI研究领域。

1989年8月，在美国底特律召开的第11届国际人工智能联合会议的专题讨论会上首次出现数据库中的知识发现(Knowledge Discovery in Database，KDD)这一术语。随后，在1991年、1993年和1994年都举行了KDD专题讨论会，汇集来自各个领域的研究人员和应用开发者，集中讨论数理统计、海量数据分析算法、知识表示和运用等问题。最初，数据挖掘是作为KDD中利用算法处理数据的一个步骤，后来逐渐演变成KDD的同义词。

KDD常常被称为数据挖掘，实际上两者是有区别的。一般将KDD中进行知识学习的阶段称为数据挖掘。数据挖掘是KDD中一个非常重要的处理步骤。但是，人们往往不加区别地使用。

数据挖掘是近年来出现的客户关系管理、商业智能等热点领域的核心技术。

5.1.1 定义

从技术角度而言，数据挖掘是从大量的、不完全的、有噪声的、模糊的、随机的实际数据中，提取隐含在其中、人们事先不知道但又是潜在有用的信息和知识的过程。这一定义包括多层含义，即数据源必须是真实的、海量的、有噪声的，发现的知识应是用户感兴趣的，并且是可接受、可理解和可运用的，并不要求是放之四海而皆准的知识，可以仅支持特定的问题。

从商业角度而言，数据挖掘是一种新的商业信息处理技术，其主要特点是对数据库中的大量业务数据进行抽取、转换、分析和其他模型化处理，从中提取辅助商业决策的关键性信息和知识。

简言之，数据挖掘本质上是一种深层次的数据分析方法。因此，数据挖掘可以描述为按企业既定业务目标，对大量的数据进行探索和分析，揭示隐藏的、未知的或验证已知的规律

性，并进一步将其模型化的有效方法。

数据挖掘所发现的最常见的知识包括：

(1) 广义(Generalization)知识

广义知识是类别特征的概括性描述知识。根据数据的微观特性发现其表征的、带有普遍性的、较高层次概念的、中观和宏观的知识，反映同类事物的共同性质，它是对数据的概括、精炼和抽象。

广义知识的发现方法和实现技术有多种，如数据立方体、面向属性的归约等。其中，数据立方体的基本思想是实现某些常用的代价较高的聚集函数的计算，如计数、求和、平均值和最大值等，并将其存储在多维数据库中。因为很多聚集函数都需要重复计算，在多维数据立方体中存放预先计算好的结果将保证快速的响应，并灵活地提供不同角度和抽象层次上的数据视图；另一种广义知识的发现方法是加拿大 Simon Fraser 大学提出的面向属性的归约方法。该方法以类 SQL 语言表示数据挖掘查询，收集数据库中的相关数据集，然后应用一系列数据分析技术，包括属性删除、概念树提升、属性阈值控制、计数及其他聚集函数计算等。

(2) 关联(Association)知识

关联知识是反映一个事件和其他事件之间依赖或关联的知识。如果两项或多项属性之间存在关联，则其中的一项就可以依据其他属性值进行预测。最著名的关联分析算法是 R. Agrawal 提出的 Apriori。关联分析的实现步骤是第一步通过迭代识别所有的频繁项集，要求频繁项集的支持率不低于用户设定的阈值；第二步从频繁项集中构造可信度不低于用户设定的阈值的规则。该算法的核心是识别或发现所有频繁项集，也是计算量最大的部分。

(3) 分类(Classification&Clustering)知识

分类知识是反映同类事物共同性质的特征知识和不同事物之间差异性的特征知识。最典型的分类方法是基于决策树的分类，它是从实例集中构造决策树，是一种有监督和指导的学习方法。该方法先根据训练集(又称为窗口)形成决策树。如果该决策树不能对所有样本给出正确的分类，则选择一些例子加入到窗口中，重复该过程直到形成正确的决策集，即一棵决策树，其叶结点是类名，中间结点是带有分枝的属性，分枝对应该属性的某一可能值。最经典的是 ID3 算法，采用自顶向下不回溯策略，保证找到一棵简单的树。C4.5 和 C5.0 等算法都是 ID3 的扩展，将分类从类别属性扩展到数值型属性。

此外，还有统计、粗糙集(Rough Set)和神经网络等分类方法。

(4) 预测(Prediction)知识

预测知识根据时间序列，由历史的和当前的数据预测未来，也可以认为是以时间为关键属性的关联知识。

目前，时间序列预测的主要方法包括统计、神经网络和机器学习等。1968 年 Box 和 Jenkins 提出了一套比较完善的时间序列建模理论和分析方法，通过建立随机模型，如自回归模型、自回归滑动平均模型、求和自回归滑动平均模型和季节调整模型等，进行时间序列预测。由于大量的时间序列是非平稳的，其特征参数和数据分布随着时间的推移而发生变化。因此，仅仅通过对某段历史数据的训练，建立单一的预测模型，还无法准确地预测。为此，人们提出了基于统计学和精确性的再训练方法，当发现现有预测模型不再适用于当前数据时，对模型重新训练，获得新的参数，建立新的模型。也有许多系统借助并行算法实现时

间序列预测。

(5) 偏差(Deviation)型知识

偏差型知识是对差异和极端特例的描述，揭示事物偏离常规的异常现象，如标准类外的特例，数据聚类外的离群值(Outlier)等。

上述知识都可以在不同的概念层次上被发现，并随着概念层次的提升，从微观到中观再到宏观，以满足不同用户不同层次的决策需要。

数据挖掘的演化过程如表5.1所示。

表5.1 数据挖掘的演化过程

进化阶段	商业问题	支持技术	产品厂家	产品特点
数据搜集(20世纪60年代)	"过去五年中总收入是多少?"	计算机、磁带和磁盘	IBM、CDC	提供历史的、静态的数据
数据访问(20世纪80年代)	"在新英格兰的分部去年三月份的销售额是多少?"	关系数据库(RDBMS)，结构化查询语言(SQL)	Oracle、Sybase、IBM、Microsoft	提供记录级的历史、动态的数据
数据仓库、决策支持(20世纪90年代)	"在新英格兰的分部去年三月份的销售额是多少?波士顿据此可得出什么结论?"	联机分析处理(OLAP)、多维数据库、数据仓库	Pilot、Comshare、Arbor、Cognos和Microstrategy等	在各种层次上提供回溯、动态的数据
数据挖掘	"下个月波士顿的销售趋势如何?为什么?"	数据挖掘算法、多处理器计算机、海量数据仓库	Pilot、Lockheed、IBM、SGI等	提供预测性信息

在了解数据挖掘演化过程的基础上，以下两个问题值得探讨。

(1) 数据挖掘与统计学的关系

近年来，人们逐渐发现数据挖掘的许多工作都是利用统计方法实现的。一些人(尤其是统计学家)甚至认为数据挖掘是统计学的一个分支，当然大多数人(包括绝大多数数据挖掘研究人员)并不这么认为。但是，统计学和数据挖掘的目标非常相似，而且数据挖掘的许多算法确实源于数理统计，统计学对数据挖掘发展的贡献功不可没。

(2) 数据挖掘与传统数据分析方法的区别

数据挖掘的本质是一种深层次的数据分析方法。数据分析已有多年的历史，只不过过去数据收集和分析的一般目的是用于科学研究。另外，由于当时计算能力的限制，很难实现对海量数据进行非常复杂的分析。现在，由于各行业业务自动化的实现，商业领域产生了大量的业务数据，这些数据并不是为了分析的目的而收集的，而是在商业活动过程中由于业务需要而自然产生的。不再是单纯为了研究，更主要的是为商业决策提供真正有价值的信息和知识，进而使利润最大化。所有企业面临的一个共同问题是企业数据量非常大，而其中真正有价值的信息和知识却很少，因此需要对大量数据进行深入分析，获得有利于提高核心竞争力的信息和知识，如同从矿石中淘金一样，数据挖掘也因此而得名。

数据挖掘与传统数据分析方法的区别主要在于：

(1) 数据挖掘的数据源与以前相比有显著的改变。首先，数据挖掘出现的背景是"数据

爆炸而知识贫乏”,它需要处理的数据量达到了“太”(万亿)级以上,比传统数据分析处理的数据量超出几个乃至十几个数量级。对于如此大规模的数据,传统的数据分析方法可能根本无法处理,即使能够处理,效率也是一个瓶颈。因此需要对原有的数据分析方法重新检验并加以改进。其次,传统数据分析方法的数据源一般都是清洁的、结构化的,数据挖掘则是从不完全的、有噪声的、模糊的数据中发现知识。数据的抽取、清洁、转换和集成是数据挖掘的重要组成部分。数据挖掘不仅可以处理结构化的数据,还可以处理半结构化或非结构化的数据。事实上,非结构化的文本挖掘乃至半结构化的 Web 挖掘正是数据挖掘的研究方向之一。

(2) 传统数据分析方法一般都是先给出一个假设然后验证,即在一定意义上是假设驱动的;与之相反,数据挖掘在一定意义上是发现驱动的,模式都是通过大量的探索从海量数据中自动获取。这一点是数据挖掘区别于传统数据分析方法以及 OLAP 技术的本质特点。数据挖掘是在事先没有假定想法与问题的情况下,在大量数据中发现隐含的模式。所得到的信息和知识具有预先未知的特征,即数据挖掘要发现那些不能靠直觉发现的甚至是违背直觉的信息或知识,越是出乎意料,可能越有价值。在商业应用中最典型的例子就是一家连锁店通过数据挖掘发现小孩尿布和啤酒之间有着惊人的联系。

5.1.2 功能

概括地,数据挖掘的主要功能如下:

1. 概念/类别描述(Concept/Class Description)

概念/类别描述是对数据集做一个简洁的总体性描述并/或描述其与某一对照数据集的差别。

例 5.1:收集移动业务每月 ARPU(Average Revenue Per User,每户平均收入)超出300 元的客户资料,然后利用数据挖掘可作出总体描述如下:年龄 35~50 岁、工作稳定、月收入 10 000 元以上、拥有良好信用度。

例 5.2:对比移动业务每月 ARPU 超出 200 元和低于 30 元的两个客户群,然后利用数据挖掘可做出描述如下:每月 ARPU 超出 200 元的客户 80%以上年龄在 35~50 岁,月收入 6000 元以上;而每月 ARPU 低于 30 元的客户 60%以上要么年龄过大要么年龄过小,月收入 2000 元以下。

2. 关联分析(Association Analysis)

从一个数据集中发现关联规则,该规则显示给定数据集中经常一起出现的属性-值元组。例如:关联规则 $X \Rightarrow Y$ 所表达的含义是满足 X 的元组很可能满足 Y。关联分析在交易数据分析、支持定向市场、商品目录设计和其他业务决策等方面有着广泛的应用。

3. 分类和预测(Classification and Prediction)

分类是指通过分析一个类别已知的数据集的特征建立分类模型,该模型可预测类别未知对象的类别。分类模型可以表现为多种形式,如分类规则(IF-THEN)、决策树或数学公式乃至神经网络。预测与分类类似,只不过预测的既可以是类别,也可以是连续的数值。

4. 聚类分析(Clustering Analysis)

聚类分析又称为“同质分组”或“无监督的分类”,即把一组数据划分为不同的“簇”,每

一簇中的数据相似而不同簇间的数据则相异，可以通过距离函数等度量相似性。聚类应保证不同类间数据的相似性尽可能小，而类内数据的相似性尽可能大。

5. 时间序列分析(Time Series Analysis)

时间序列分析即预测，是指通过对大量时间序列数据的分析找到特定的规则和感兴趣的特性，包括搜索相似序列或子序列，挖掘序列模式、周期性、趋势和偏差。预测的目的是对未来的情况做出估计。

6. 其他功能

除了上述主要功能外，还包括偏差分析(Deviation Analysis)、孤立点分析(Outlier Analysis)等。

随着数据挖掘技术的不断发展，新的应用层出不穷。

5.1.3 模型

1. 5A 模型

5A 模型是 SPSS 提出的，强调的是数据挖掘工具应具有的功能和能力。5A 模型认为数据挖掘方法学由 5 个基本元素组成，即 Assess、Access、Analyze、Act 和 Automate。

- Assess：正确、彻底地评价任务的需求和数据，正确地理解商业问题和数据，并设计挖掘计划及相关准备任务。
- Access：方便、快速地存取任务涉及的数据，要求指定的数据集合符合挖掘的需求和质量。
- Analyze：适当、完备的分析技术和工具，要求工具能全面提供适合不同需求的各种挖掘、建模算法。
- Act：具有推荐性、说服力的模型演示，提供可视化的模型并能够灵活嵌入到各类展示平台中。
- Automate：自动地提供挖掘结果并展现给用户。

2. SEMMA 模型

SEMMA 是 SAS 提出的数据挖掘过程模型，由抽样(Sample)、探索(Explore)、调整(Modify)、建模(Model)和评价(Assess)五个步骤组成，如图 5.1 所示。

1) 数据取样

进行数据挖掘时，首先需要从大量数据中取出一个与所要探索问题相关的数据子集，而不是动用全部数据。犹如对开采出来的矿石首先要进行选矿一样。通过数据筛选，不仅能减少数据处理量，节省系统资源，而且使所反映的规律性更加凸现出来。

2) 数据特征探索、分析和预处理

前面提及的数据取样，多少是带着人们对如何达到数据挖掘目的的先验认识进行操作的。当获得样本数据集后，是否可达到预想的要求；是否存在明显的规律和趋势；因素之间有何相关性；可划分为怎样的类别等都是需要探索的内容。进行数据特征的探索和分析，最好是采用可视化的操作，显示各种统计分析的结果，而且可以多维、动态甚至旋转方式显示。

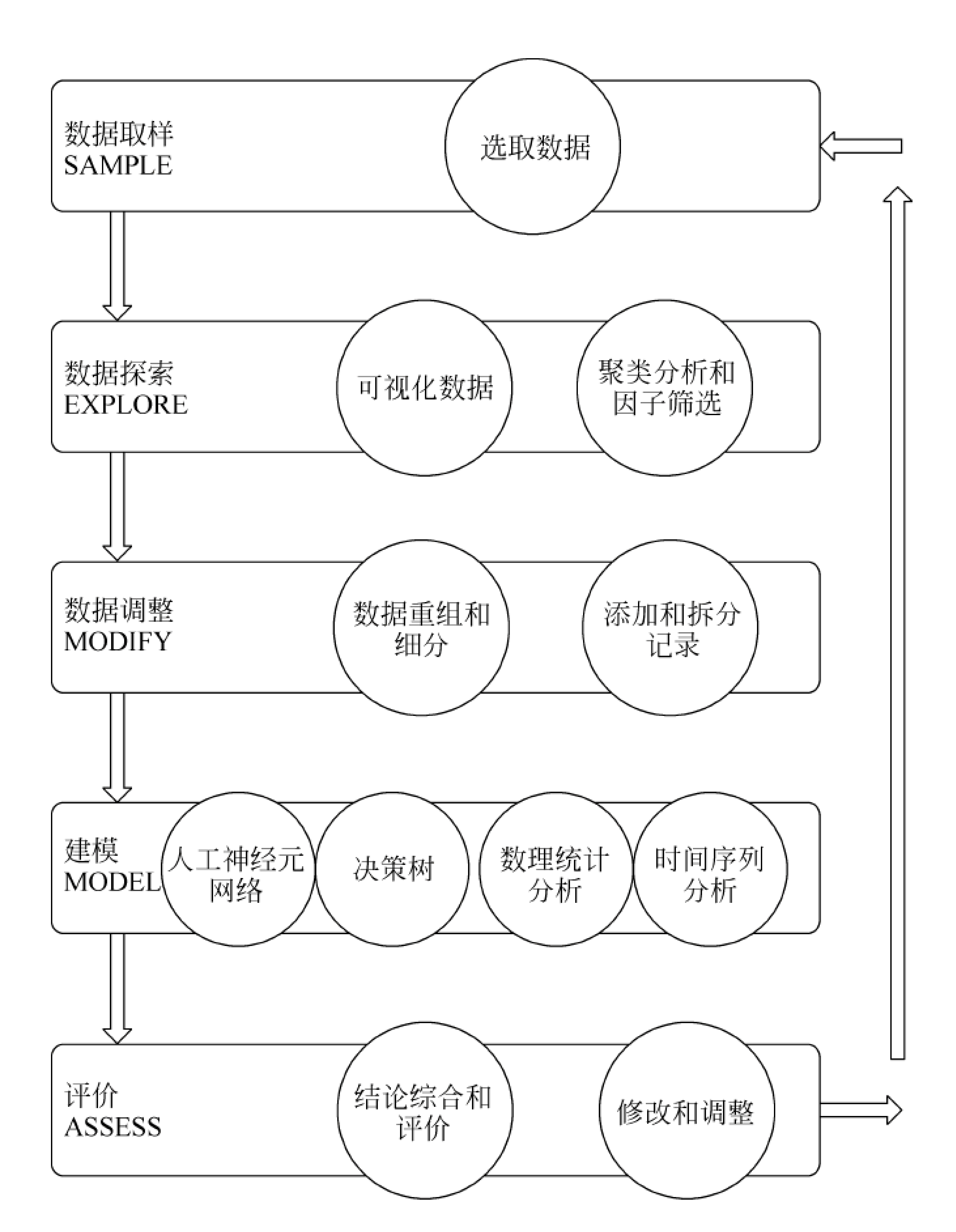


图 5.1 数据挖掘过程模型——SEMMA

3) 问题明确化、数据调整和技术选择

通过上述两个步骤,对数据的状态和趋势可能有了进一步的了解,对原先要解决的问题可能会进一步明确,这时应尽可能对需求进行量化,这是十分重要的。例如很可能因为诸如质量不好、生产率低等一些模糊的问题而无法进行有效的数据挖掘。

在问题明确的基础上,可以按照问题的具体要求审视数据集,以确定是否满足需求。Gartner Group 在评论当前一些数据挖掘产品时特别强调指出:在数据挖掘的各个阶段,数据挖掘的产品都要让所使用的数据和所建立的模型处于十分易于调整、修改和变动的状态,这才能保证数据挖掘的有效进行。

针对需要可能要对数据进行增删,也可能按照对整个数据挖掘过程的新认识,组合或者生成一些新的变量。

4) 模型的研发、知识的发现

正如 Gartner Group 评论中所指出的:数理统计方法还是数据挖掘工作中最常用的主流手段。挖掘过程中可能需要各种不同类型的模型、不同特征数据的回归分析,如正交回归、线性回归、Logistic 回归和非线性回归等,可处理的数据包括实型、有序和属性数据等,并能产生各种有用的统计量和诊断信息。

5）模型和知识的综合解释和评价

3．CRISP-DM 模型

为了使数据挖掘技术在产业界得到更好的应用，欧洲委员会联合一些数据挖掘软件厂商开发了 CRISP-DM(Cross Industry Standard Process for Data Mining)模型，如图 5.2 所示，旨在把数据挖掘过程标准化，使数据挖掘的实施速度更快、成本更低、更可靠并且更易于管理。1996 年 CRISP-DM 模型被首次提出，并在各种 KDD 模型中占据领先地位，份额达到近 60%。

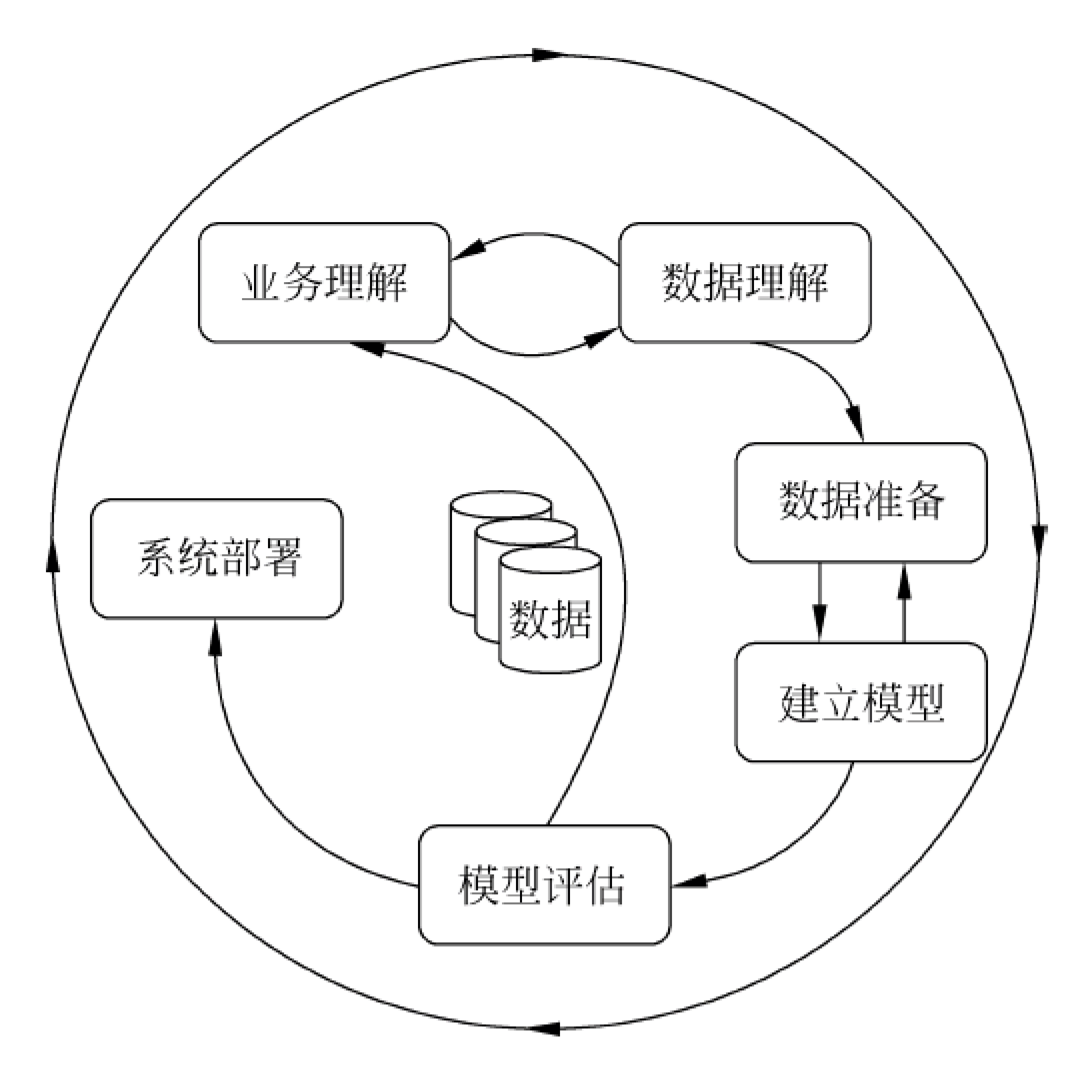


图 5.2　CRISP-DM 模型

CRISP-DM 模型中，数据挖掘过程划分为如下六个阶段。

1）业务理解(Business Understanding)阶段

本阶段专注于从商业角度理解项目目标和需求，并转化为数据挖掘的问题定义，同时设计出一个初始计划。具体包括：

(1) 确定业务目标(Determine Business Objectives)

分析员第一步要做的是从商业角度全面地理解客户真正希望达到的目的。客户的目标常常具有挑战性但同时又有很多限制，需要很好地权衡。分析员最开始要做的是找到影响整个项目的重要因素。如果缺少这一阶段，很可能会导致给出的答案是正确的但是提出的问题却是错的。这一步骤应获得的相关细节包括：

- 背景(Background)——了解有关组织或企业商业状况的详细信息。
- 商业目标(Business Objectives)——从商业角度描述客户的最主要目标，客户通常应该从商业角度提出很多问题，例如一个主要的商业目标是预测客户何时可能流向竞争对手，提出的问题可能就是“一个电信客户所使用的主要业务受理渠道(ATM、客户经理、Internet)对他们是否流失是如何起影响作用的?”或是“降低 ATM 交易费能否减少高价值客户的流失?”
- 商业成功标准(Business Success Criteria)——从商业角度描述一个数据挖掘项目的结果是否成功。可能是以一种明确的方式如减少客户流失率至某个级别或者更一般更主观的方式如“发现有用的联系”，对于后者应该注明是谁做出的这一主观决定。

(2) 评估商业环境(Assess Situation)

这一任务涉及对很多资源、约束、假定等对整个项目计划和目标有影响的因素的仔细考察。前一个任务只是快速了解大致情况。这一步骤应获得的相关细节包括：

- 资源列表(Inventory of Resources)——列出与项目相关的全部资源,包括人(领域专家、数据专家、技术支持、挖掘人员)、数据(固定抽取的数据、从数据仓库或业务数据库中获得的数据)、计算资源(硬件平台)和软件资源(数据挖掘工具及其他相关资源)。
- 需求、假定、约束(Requirement, Assumption and Constraint)——列出项目的全部需求,包括完工日程、对结果的质量要求、安全性以及法律相关问题。列出项目所作的假定以及约束,如对资源的约束或对技术的约束,如数据的大小对于建模是可行的。
- 风险和意外开支(Risk and Contingency)——列出可能导致项目延期或终止的事件,列出如果发生风险所带来的意外开支。
- 术语表(Terminology)——一个与项目有关的术语表应该有两个：一是商业术语表,另一个是数据挖掘的技术术语表。
- 成本和收益(Cost and Benefit)——项目的成本收益分析,比较项目的潜在收益及其耗费成本。这项工作应该非常具体,如采用商业上的货币度量方式。

(3) 确定数据挖掘目标

商业目标用商业术语说明,数据挖掘目标则用技术术语说明。如商业目标是“提高已有客户的销售量”而数据挖掘目标可能是“在给定其过去三年的购买情况数据、人口信息(年龄、收入和所在城市等)和商品价格条件下,预测某个客户可能会购买多少”。这一步骤的相关细节包括：

- 数据挖掘目标(Data Mining Goal)——描述项目预期的哪些输出会使商业目标获得成功。
- 数据挖掘成功标准(Data Mining Success Criteria)——用技术术语定义项目成功时的结果。如果商业目标是以主观方式描述的则此处的描述可能也将以主观术语描述,但做出描述的人应该被注明。

(4) 提出项目计划(Produce Project Plan)

描述一个可行的计划以达到数据挖掘目标和商业目标。计划应该预先明确项目的余下步骤包括工具和技术的初步选取。这一步骤的相关细节包括：

- 项目计划(Project Plan)——列出项目的执行阶段,包括时间、所需资源、输入、输出和依赖关系,可能会在某些环节循环或重复,项目的计划根据实际需要动态更新。
- 初步估计工具和技术(Initial Assessment of Tool and Technique)——对使用的工具和技术做出初步估计和计划。

2) 数据理解(Data Understanding)阶段

本阶段,先收集初步的数据,然后了解并熟悉数据,以识别数据质量、找到对数据的基本观察或假设隐含的信息以检测出感兴趣的数据子集。具体包括：

(1) 收集原始数据(Collect Initial Data)

获取在项目资源中所需的原始数据,这个过程可能包括数据理解,这相当于初步的数据

准备过程。如果数据源分散则需要对数据进行集成，生成原始数据收集报告(Initial Data Collection Report)，其中列出获得的数据集，列出其在项目中的位置、获取数据的方法和遇到的任何问题及其解决方法。

(2) 描述数据(Describe Data)

查看数据的表面特性，生成数据描述报告(Data Description Report)，描述数据的格式、质量、字段数、记录数以及其他各种表面特性。

(3) 探索数据(Explore Data)

该任务为进行数据挖掘进行一定的查询、可视化以及报告工作。包括分析关键属性，预测任务的目标属性、属性间的关系、简单聚集的结果、重要子集的特征和简单统计的分析。这些分析可能直接对数据挖掘目标产生影响也可能对数据描述、数据质量报告或数据准备有用。生成数据探索报告(Data Exploration Report)描述此项工作的结果，包括最初的发现、原始假设及其对项目的影响。该报告可能有一些图等表示数据的特性或有意义的子集。

(4) 检查数据质量(Verify Data Quality)

检查数据质量，可以提出类似于数据是否完整(是否覆盖全部情况)，是否正确，如果有错误，错误率是多少，数据中是否有缺失值，如果存在缺失值则它们是如何表示的，在何处发生及发生率是多少等问题。生成数据质量报告(Data Quality Report)，列出数据质量验证结果，如果存在质量问题列出可能的解决方案，解决方案同时依赖于数据以及商业知识。

3) 数据准备(Data Preparation)阶段

本阶段包括从数据构造到最终数据集合(将要输入建模工具的数据)的所有活动。数据准备任务可能需要执行很多次，并没有任何规定的顺序。任务包括表、记录属性的选择以及为适应建模工具的要求对数据的转换和清洗。具体包括：

(1) 数据选择(Select Data)

选择分析用的数据。数据选择的标准是与挖掘目标、质量和技术约束相关。数据的选择包括行和列的选择。生成所包含/不包含的数据清单(Rationale for Inclusion/Exclusion)，列出所包含/不包含的数据清单及其原因。

(2) 数据清洁(Clean Data)

提高数据质量以达到数据挖掘的要求。可能包括选择已经清洁的数据子集，对于错误数据的修正等。生成数据清洁报告(Data Cleaning Report)描述应该采取哪些方法和措施解决数据质量问题，这些问题在验证数据质量报告中曾经提出。

(3) 数据创建(Construct Data)

该任务指根据需要产生新的派生属性、新的记录或变换已有属性值。这一步骤的相关细节包括：

- 派生属性(Derived Attribute)——将已有属性组合或变换成更有利于知识挖掘的派生属性，如面积＝长×宽。
- 生成记录(Generate Record)——说明新生成的数据，例如客户去年没有购买记录，其有关购买数量的记录不存在，创建一条记录将其购买数量设置为 0 以利于分析。

(4) 数据合并(Integrate Data)

使用一些方法将多个表或记录合并为新的记录或值。包括合并数据(Merged Data)，合并表(将两个或多个对同一对象有不同信息的表中的记录合成在一张表中)。合成数据也包

括聚集,即将若干记录的信息累计生成新的信息。

(5) 数据格式化(Format Data)

数据格式化指可能因为建模工具的要求改变数据的形式。这一步骤的相关细节包括重新格式化数据(Reformatted Data):有些工具对属性的顺序有要求,例如第一个域是记录标识符,最后一个域用于预测模型的输出。记录的顺序同样很重要,可能某些工具需要记录根据输出值排序,通常的情况下表的记录是有顺序的,但算法要求使用乱序的数据,例如神经网络在记录是随机排列时效果最好,不过通常无须人为干预而由工具本身完成。除此之外,还有一些数据有按词法格式化的需求,如对于逗号分割的文本字段可能需要删去逗号,字符串最长不超过 32 个字符等。

4) 建立模型(Modeling)阶段

本阶段可以选择各种建模技术,各类模型参数也可以调整优化。对同一个数据挖掘问题可以有多种可用技术,某些技术对数据的形式有一定的要求,因此常常要退回到数据准备阶段。具体包括:

(1) 选择建模技术(Select Modeling Technique)

确定数据挖掘算法和参数,可能会利用多个算法。作为建模任务的第一步,应该选择实际使用的建模技术。与在商业理解阶段选择建模技术相比本阶段更加明确指定具体的建模技术,如明确决策树是使用 C4.5 算法构造还是 BP(Back Propagation)神经网络算法。如果使用多种技术,需要对每一种技术分别进行此项工作。选择什么样的模型决定了需要对数据做哪些预处理,如神经网络模型需要转换数据,有些数据挖掘工具可能对输入数据的格式有特定限制。一旦所有的数据准备好之后,可开始训练模型。这一步骤除建模技术外还包括建模假设(Modeling Assumption),即很多建模技术对数据都做一定的假设,例如所有属性具有同样的权重,没有缺失值,分类属性必须是符号化的等。

(2) 测试方案设计(Generate Test Design)

设计某种测试模型的质量和有效性的机制。在真正生成模型之前,需要建立一个测试过程或机制来保证模型的质量和正确性,例如对于分类很自然地使用错误率作为分类模型的质量评价度量。因此可以将数据分为训练集和测试集,在训练集上建立模型,在测试集上测试模型的质量。这一步骤的相关细节包括:

- 测试方案(Test Design)——该方案说明如何训练、测试和评估模型,其中一个主要部分是数据如何划分为训练集、测试集和评估集。
- 模型训练(Build Model)——在准备好的数据集上运行数据挖掘算法,得出一个或者多个模型。
- 模型测试评估(Assess Model)——根据测试方案进行测试,从数据挖掘角度确定数据挖掘目标是否成功。

5) 模型评估(Evaluation)阶段

在最终扩展模型前需要彻底地评价模型,对所建模型再次考察其执行步骤并确信其正确地达到了商业目标。这里,一个关键的目的是确定是否有某些重要的商业问题还没有充分地考虑到。具体包括:

- 结果评估(Evaluate Result)——从商业角度评估获得的模型,甚至实际试用该模型测试其效果。

• 过程回顾(Review Process)——回顾项目的所有流程,确定每一阶段都没有失误。
• 确定下一步工作(Determine Next Step)——根据结果评估和过程回顾得出的结论,确定是部署该挖掘模型还是从某个阶段重新开始。

6) 部署(Deployment)阶段

所获得的挖掘结果和知识应该采用用户可以使用的方式来组织和表示。可以简单到只有一份报告也可以实现一个可以重复的挖掘过程或系统。很多情况下,这将由用户而非分析员实施。具体包括:

• 部署计划(Plan Deployment)——对在业务运作中部署模型做出计划。
• 监控和维护计划(Plan Monitoring and Maintenance)——如何监控模型在实际业务中的使用情况,如何维护该模型。
• 做出最终报告(Produce Final Report)——总结项目经验和项目结果。
• 项目回顾(Review Project)——回顾项目的实施过程,总结经验教训,预测数据挖掘的运行效果。

为了保证数据挖掘项目的可靠性和可管理性,按照CRISP-DM模型规定应该产生11个报告,即业务理解报告、原始数据收集报告、数据描述报告、数据探索报告、数据质量报告、数据集描述报告、模型训练报告、模型评估报告、部署计划、监控和维护计划和总结报告。通过这些报告,可以有效地控制数据挖掘的项目进程,减少开发风险。

CRISP-DM模型给出了整个数据挖掘项目的流程。对于一个数据挖掘系统,实际上只涉及到其中的数据准备、模型训练和模型评估三个阶段,而业务理解和数据理解则是为数据挖掘系统准备输入,模型部署是数据挖掘系统的实施。

5.1.4 展望

近年来,数据挖掘的研究重点逐渐从发现方法转向系统应用,注重多种策略和技术的集成,以及多学科之间的相互渗透,如1998年在美国纽约举行的第四届知识发现与数据挖掘国际学术会议上不仅进行了学术讨论,并且有三十多家软件公司展示了数据挖掘软件产品,不少软件已在北美、欧洲得到应用。

目前,数据挖掘在银行、电信、保险、交通和零售(如超市)等领域都有成功的应用案例,随着商业竞争加剧,对数据挖掘的需求将愈加紧迫。

就应用领域而言,当前数据挖掘的热点包括网站的数据挖掘、生物信息或基因(Bioinformatics/genomics)挖掘、文本挖掘(Text Mining)和多媒体挖掘等。下面分别加以简要介绍。

1. 网站的数据挖掘

当前Internet上各类电子商务网站风起云涌,电子商务的竞争比传统商业竞争更加激烈。客户从一个电子商务网站转换到竞争对手那边,只需单击几下鼠标即可,电子商务环境下客户保有比传统商业更加困难,若想在激烈的竞争中生存,则必须比竞争对手更了解客户。电子商务网站每天都可能有上百万次的在线交易,生成大量的日志文件(Log File)和登记表,如何对这些数据进行分析和挖掘,充分了解客户的喜好、购买习惯甚至是客户一时的冲动,设计出满足不同客户群需求的个性化网站,进而增加竞争力,几乎是势在必行。

就分析和建模而言,网站的数据挖掘和现有的数据挖掘差别并不是特别大,很多方法和分析思路都可以借鉴,所不同的是网站的数据格式有很大一部分来自于点击流,与传统的数据格式有所区别。因而对电子商务网站进行数据挖掘所做的主要工作是数据准备。目前,有很多厂商正致力于开发专门的软件。

2. 生物信息或基因挖掘

生物信息或基因挖掘则完全属于另一个领域,其商业价值很难估计,但对于人类却受益匪浅。例如,基因组合千变万化,患有某种疾病的人的基因和正常人的基因到底差别多大?能否找出差异,进而加以改变使之成为正常基因?这都需要数据挖掘技术的支持。

对于生物信息或基因的挖掘和通常的数据挖掘相比,无论在数据的复杂度、数据量以及分析和建模上都复杂得多。就算法而言,需要一些新的和好的算法,目前很多厂商致力于这方面的研究;就技术和软件而言,还远没有达到成熟。

3. 文本挖掘(Text Mining)

文本挖掘是另一个人们颇感兴趣的领域,例如客户服务中心把同客户的谈话内容转换为文本,再进行挖掘,进而了解客户对服务的满意度和客户的需求以及客户之间的相互关系等。

无论是在数据结构还是在分析方法方面,文本挖掘和前面提及的数据挖掘相差很大。文本挖掘并不是一件容易的事情,尤其是在分析方法方面,还有很多需要研究的专题。目前市场上有一些类似的软件,但大部分只是把文本移来移去,或简单地计算某些词汇出现的频率,并没有真正的分析功能。

4. 多媒体挖掘(Multimedia Mining)

多媒体挖掘主要包括两个方面。基于描述的检索是指基于图像描述创建索引并实现对象检索,如关键字、标题、尺寸和创建时间等。若通过人工实现极为费力,若自动实现往往结果不甚理想;基于内容的检索是指支持基于图像内容的检索,如颜色、质地、形状和对象等。

当前,数据挖掘研究的焦点集中在以下几个方面:

(1) 发现语言的形式化描述

即研究专门用于知识发现的数据挖掘语言,寻求类似于数据库 SQL 语言的数据挖掘语言,使挖掘过程走向形式化和标准化。

(2) 寻求数据挖掘过程中的可视化方法

使知识发现的过程能够被用户理解,便于在知识发现过程中实现人机交互。

(3) 研究在网络环境下的数据挖掘技术

特别是在 Internet 上建立 Web 服务器,并且与数据库服务器配合,实现 Web 挖掘(Web Mining)。

(4) 加强对各种非结构化数据的挖掘(Data Mining for Audio&Video)

如对文本、图形、视频、音频乃至流媒体的挖掘。

(5) 知识的维护更新

数据挖掘的结果——知识是具有时效性的,需要研究知识的维护更新技术,如知识的增量更新以及模型进化等。

5.2 实现

数据挖掘的实现过程如图 5.3 所示。

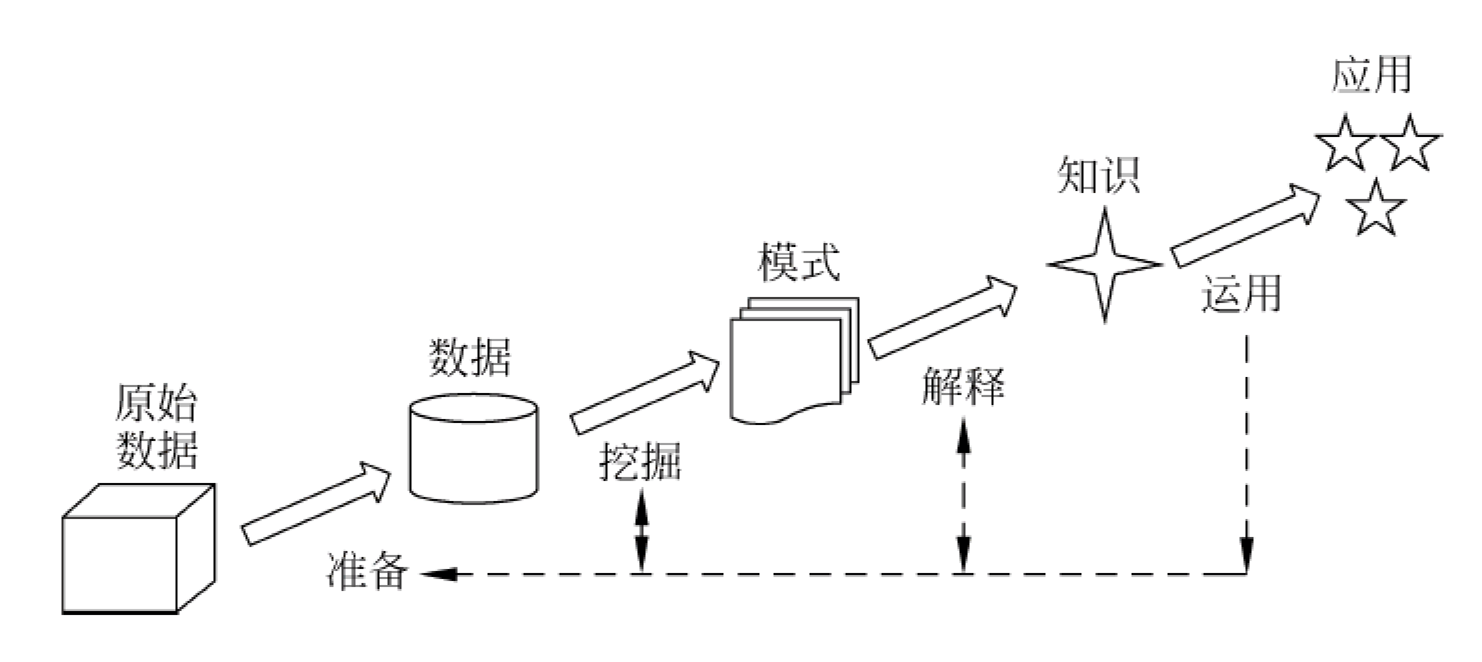


图 5.3 数据挖掘的实现过程

1. 数据准备

KDD 的处理对象是海量数据，它们一般存储在数据库系统，是长期积累的结果。但往往不适合直接进行数据挖掘，需要进行预处理。数据预处理包括数据的选择、清洁（消除噪声、冗余数据）、推测（推算缺失数据）、转换（离散型数据与连续型数据之间的转换）、数据缩减（减少数据量）等。

数据准备是 KDD 的第一个步骤，其成功与否将影响到数据挖掘的效率、准确性以及最终模型的有效性。

2. 数据挖掘

数据挖掘是最为关键的一个步骤，它根据 KDD 的目标，选取相应算法的参数，分析数据，得到可能形成知识的模型，目前常用的包括决策树、分类、聚类、粗糙集、关联规则、人工神经网络和遗传算法等。

3. 模式的评估、解释

通过上述步骤得到的模式，有可能是没有意义或没有实用价值的，因此需要评估，确定哪些是有效的、有用的模式。此外，大部分模式是数学的表达式，很难被人理解，还需要将其解释成可理解的方式呈现给用户。

4. 知识运用

发现知识是为了运用，如何运用知识也是 KDD 的关键。运用知识主要有两种途径：其一是只需看知识本身描述的关系或结果，就可以对决策提供辅助支持；其二是要求对新的数据运用知识，由此可能产生新的问题，并需要对知识做进一步优化。

KDD 过程可能需要多次的循环反复，一旦某个步骤与预期目标不符，则需要回溯到前面的步骤，重新调整和执行。

在数据挖掘的实现过程中，不同步骤需要具有不同专长的人员参与完成，大致可以分为三类：

- 业务分析人员——要求精通业务，能够确定数据定义和挖掘算法的业务需求。
- 数据分析人员——精通数据分析技术，熟练掌握统计学方法，有能力把业务需求转化为数据挖掘的各步操作，并为每步操作选择合适的技术。

• 数据管理人员——精通数据管理技术,负责从数据库或数据仓库中获取数据。

由此可知,数据挖掘是一个多领域专家合作的过程,同时也是一个在资金和技术上高投入的过程,这一过程需要反复进行,以不断地趋近事物的本质,不断地优化问题的解决方案。

一般地,企业实施数据挖掘的方式主要是三种,即:

• 购买成熟的模型
• 购买通用的数据挖掘软件
• 定制数据挖掘系统

第一种方式实现简单,可以直接应用,但是要求模型所模拟的环境必须和企业的产品、客户以及市场条件相类似。当模型涉及的环境条件改变时,不能根据环境的变化做出修改和调整。第二种方式可以利用数据挖掘系统根据企业自身的数据生成模型。但是,一个通用的数据挖掘系统在对特定的商业问题的理解上可能需要做很多工作。同时,如何与企业现有系统自动化集成也是一个需要着重考虑的问题。第三种方式可以较好地解决与现有系统集成的问题,并可以直接面向特定的商业问题的解决。但是这种方式实现较复杂,项目实施周期长,成本较高。

当然,企业也可以把上述三种方式结合起来,例如购买包含模型的数据挖掘软件、购买通用数据挖掘软件系统进行针对本企业的二次开发等。

根据电信行业的特点,我们认为构建一个适合企业自身特点的数据挖掘系统是较好的选择(包括购买针对本行业特点开发的数据挖掘系统)。

5.3 工具

5.3.1 概述

数据挖掘的应用前景非常广阔,相关产品研发方兴未艾。目前,已经出现了几十种商用数据挖掘产品和工具。

数据挖掘工具的发展大体经历了四个阶段。

第一代数据挖掘工具支持一个或少数几种数据挖掘算法,可以挖掘向量数据(Vector-valued Data),挖掘时一般一次性调入内存进行处理。典型产品包括 CBA 和 Salford Systems 公司早期的 CART 系统。

第二代数据挖掘工具支持数据库和数据仓库,和它们之间有高性能接口,具有较好的可扩展性,能够挖掘更大、更复杂以及高维的数据集。通过支持数据挖掘模式(Data Mining Schema)和数据挖掘查询语言(DMQL)增加了系统的灵活性。典型产品是 DBMiner。

第三代数据挖掘工具的特点是能够挖掘 Internet/Extranet 的分布式和高度异构的数据,并且能够有效地与操作型系统集成。其关键技术之一是提供对建立在异构系统上的多个预测模型以及管理这些预测模型的元数据提供第一级的支持,但不支持移动环境。典型产品是 SPSS Clementine。

第四代数据挖掘工具能够挖掘嵌入式系统、移动系统和普适的计算设备产生的各种类型的数据。

数据挖掘工具的应用主要分为三类：

(1) 通用数据挖掘工具，不区分具体数据的含义，采用通用的挖掘算法，处理常见的数据类型。通用数据挖掘工具可以实现多种模式的挖掘，挖掘什么、用什么挖掘都由用户根据实际进行选择，如 SAS Enterprise Miner、IBM Intelligent Miner、UnicaPRW、SPSS Clementine、SGI MineSet、Oracle Darwin 和 Angoss KnowledgeSeeker 等。

(2) 综合数据挖掘工具，能提供管理报告、在线分析处理和普通的数据挖掘能力，如 Cognos Scenario 和 Business Objects 等。

(3) 专用数据挖掘工具，是面向特定应用的数据挖掘工具，针对某一特定领域的问题提供解决方案。在设计算法时充分考虑了数据和需求的特殊性，并进行优化。其针对性比较强，只能用于一种应用，因此往往采用特殊的算法，可以处理特殊的数据，实现特殊的目的，发现的知识可靠度较高。如 KDI(零售)、Options & Choices(保险)和 HNC(欺诈行为探查)等。

各种数据挖掘工具各有千秋，适用不同的环境，了解这些工具的特性，并根据企业特点选择合适的数据挖掘工具是一个非常具有挑战性的问题。由于各公司的背景、财力和挖掘水平各不相同，对数据挖掘工具的需求也不尽相同。目前，国际上在数据库、数据仓库的性能评测方面最权威的机构是交易处理性能委员会(Transaction Processing Performance Council，TPPC)。但是，到目前为止数据挖掘方面可供参考的权威评估报告非常少。最近的一份完整而权威的数据挖掘工具评估报告是由 John F. ElderIV 和 Dean W. Abbott 在 1998 年完成的，可以说它已经过时了。

一般而言，对数据挖掘工具的选择可考虑以下方面：

(1) 公司的数据挖掘需求是短期还是长期

如果是短期行为，可购买那些能解决特定问题的软件包或外包给咨询公司；如果是长期使用，需要购买功能较丰富，使用较方便，维护升级较好的企业型数据挖掘工具。

(2) 公司的数据挖掘经验和水平

应该根据公司内部数据挖掘团队的经验和水平，选取一些经过基本培训后就能掌握的工具，而不是盲目求好，最终导致因不会使用工具而将其束之高阁，从而造成不必要的浪费。在选择数据挖掘工具前，必须对公司现有的数据进行评估。如果不具备针对业务主题进行挖掘(例如风险预测)的数据或者现有格式不能满足数据挖掘工具的需求，则需要在数据具备后，再考虑购买。

(3) 公司的预算

当然，在评估数据挖掘工具时，公司也需要结合自身的财务预算做出决定。

(4) 工具的性能

好的工具可以更有效地挖掘出准确、高价值的信息和知识，所以工具性能的评估也是相当重要。

5.3.2 比较

下面将从运行平台、易用性、算法、灵活性(算法参数、选项等)和挖掘过程自动化等方面，对表 5.2 列出的常见的数据挖掘工具进行比较。

表 5.2　常见的数据挖掘工具

产　　品	公　　司	主　　页	版　　本
SPSS Clementine	Integral Solutions, Ltd.	http://www.isl.co.uk/clem.html	4.0
Darwin	Thinking Machines, Corp.	http://www.think.com/html/products/products.htm	3.0.1
Enterprise Miner	SAS Institute	http://www.sas.com/software/components/miner.html	Beta
Intelligent Miner	IBM	http://www.software.ibm.com/data/iminer/	2
PRW	Unica Technologies, Inc.	http://www.unica-usa.com/prodinfo.htm	2.1
Scenario	Cognos	http://www.cognos.com/busintell/products/index.html	2

1. 支持的平台和数据库连接

数据挖掘工具所支持的平台和数据库连接是影响其性能的重要因素。一个只有 PC 单机版的工具在处理大数据时会存在严重的效率问题,而一个具有跨平台的 C/S 架构的工具可能会具有更好的伸展性(Scalability)。而从数据库中存取数据的效率更是每个数据挖掘工具必须考虑的重要问题。常见的数据挖掘工具在平台和数据库连接方式上的比较如表 5.3 所示。

表 5.3　常见的数据挖掘工具在平台和数据库连接方式上的比较

产　　品	单 机 版	C/S　　版	数　据　源
SPSS Clementine	PC 和 UNIX	无	ODBC
Darwin	无	UNIX Server/PC Client	ODBC
Enterprise Miner	PC	UNIX Server/PC Client 和 NTServer/PC Client	ODBC 和 Native Database Drivers
Intelligent Miner	PC	UNIX Server/PC Client	Native Database Drivers
PRW	PC	无	ODBC
Scenario	PC	无	仅支持数据库文件(Dbase、excel、csv 等)

可以看到这些工具一般都有 PC 上的单机版,但并不是每一个都有 UNIX 平台的 Server 端。而且虽然它们都可以与数据库相连接,但连接方式各不相同,可能会存在很大的效率差别(更具体的连接手段可查看相关产品手册)。

2. 算法

算法是数据挖掘工具的核心,是区别于其他数据分析工具的主要因素。一个数据挖掘工具支持的算法在很大程度上体现了其性能。各种数据挖掘工具支持的主要算法如表 5.4 所示。

表 5.4 各种数据挖掘工具支持的主要算法

算　　法	SPSS Clementine	Darwin	Enterprise Miner	Intelligent Miner	PRW	Scenario
决策树	有	有	有	有	无	有
神经网络	有	有	有	有	有	无
回归分析	有	无	有	有	有	无
径向基函数(Radial Basis Function,RBF)	无	有	无	有	有	无
最近邻	无	无	有	无	有	无
最近均值	无	无	无	无	有	无
Kohonen 自组织映射	有	无	有	无	无	无
聚类	有	无	无	有	有	无
关联规则	有	无	无	有	无	无

表 5.4 中列出的只是所支持算法的大致类别,事实上还有其他算法(如时间序列等)。而且,对于同一类算法各个工具采用的具体算法也不相同。由于采用的算法不同,它们表现出不同的效率和结果。

在算法参数控制和扩展功能方面,对比上述工具可以发现 Enterprise Miner 和 PRW 对参数控制实现得较好,而这方面 Intelligent Miner 较弱。几乎所有的工具都提供对决策树实数值的处理和图形展示等功能,但只有 SPSS Clementine 和 Scenario 较好地实现了决策树的修剪功能。此外,神经网络的功能扩展方面也差别较大。

3. 易用性

作为商用产品,易用性是提高客户满意度和市场占有率的重要因素。表 5.5 从数据装载和操作、模型建立、模型理解和技术支持四个方面对上述常见数据挖掘工具进行了比较。

表 5.5 数据挖掘工具在易用性方面的比较

产　　品	数据装载和操作	模型建立	模型理解	技术支持	总体感觉
SPSS Clementine	+++	+++	+++	+++	+++
Darwin	++	++	+++	++	++
Enterprise Miner	++	++	++	++	++
Intelligent Miner	++	++	++	++	++
PRW	+++	+++	+++	+++	+++
Scenario	++	+++	+++	++	+++

注:+(较好),++(好),+++(非常好)

显然它们都具有较好的易用性,但还是有些差别。

SPSS Clementine、Enterprise Miner 和 PRW、Scenario 可以自动读入数据的第一行以决定域名和数据类型,Darwin 必须在一个说明文件中注明,而 Intelligent Miner 则提供对话框输入。各种工具都提供对列的操作(如创建新列、合并列等)和对行的操作(如取样、测试集和训练集的划分等)。

与研究性的平台相比,可视化是商用数据挖掘工具着重考虑的方面。Intelligent Miner、Enterprise Miner 和 Scenario 都具有图形化的展示,而 SPSS Clementine 和 Darwin 则提供基于文本的规则说明。此外,对于柱状图、饼图和曲线等其支持程度也各不相同。

4. 挖掘自动化

这里的挖掘自动化是指产品以何种手段使用户完成数据源选择、数据转换、算法选择和结果保存等一系列步骤之间的衔接。各种常见数据挖掘工具支持的手段如表 5.6 所示。

表 5.6 常见数据挖掘工具挖掘自动化的比较

产　　品	对自动化的支持
SPSS Clementine	支持可视化编程和编程语言
Darwin	支持编程语言
Enterprise Miner	支持可视化编程和编程语言
Intelligent Miner	仅提供向导界面,不支持编程
PRW	有一个管理组件,支持宏
Scenario	自动化支持较弱,很多过程需手工完成

SPSS Clementine 和 Enterprise Miner 支持的可视化编程是指利用拖拉拽小图标并建立连接的手段描述整个过程,而 IBM 的 Intelligent Miner 则提供一个向导使用户在每个步骤上做出选择。

表 5.6 给出的并不是所有可能的挖掘自动化支持手段,事实上有许多其他工具包含强大的脚本支持(CART、S-Plus 等),S-Plus 甚至支持 C/C++编程。

上述常用的数据挖掘工具各自适用不同的环境。IBM 的 Intelligent Miner 在市场上比较领先并提供良好的技术支持;SAS 的 Enterprise Miner 明显地偏向统计(因此更适用于统计环境);在不清楚哪种算法更好的情况下 UnicaPRW 是较好的选择;Cognos Scenario 则是其数据仓库系列产品的重要组件。

第6章 聚类分析

聚类分析(Clustering Analysis)是依据事物的某些属性将其聚集成类。使类间相似性尽量小,类内相似性尽量大,即“物以类聚,人以群分”。

聚类不同于分类(Classification),聚类是一种无监督的学习,无须任何先验知识,直接从数据对象中发现有意义的结构,输入对象被划分到一个未知的类;分类则是一种有监督的学习,依据已知的属性值对对象进行分类,输入对象被划分到一个已预先定义的类。

聚类一直是生物学、心理学、医学、考古学、图像处理、市场营销、机器学习、模式识别、数据挖掘及遥感等众多工程和技术领域的研究热点。此外,聚类还可以用作独立的数据挖掘工具,以了解数据的分布,或作为其他数据挖掘算法的预处理步骤等。

聚类分析是一个古老的问题,它伴随着人类社会的发展而不断深入。最初,聚类是统计学的一个分支,如经典的多元统计等。随后,提出了依据对象属性值的相似度实现聚类的相似聚类法。相似聚类法又分为系统聚类(预先不确定类别数目)和动态聚类(预先已确定类别数目)。相似聚类法有着广泛的应用,但同时也存在一些不足之处。从本质上说,相似聚类法属于上下文无关的聚类,即对象间的相似性度量完全依赖于对象本身的属性,并不受其上下文的影响,这对于“静态”数据的聚类是可行的,但对于“动态”数据,则聚类结果无法令人满意。继而,提出了环境聚类法,旨在将人们的注意力从客体的相似性推广到相似性以外,尽管所得到的类提供了更多的信息,但环境聚类法与相似性聚类法一样,都是概念无关的,即聚类结果不易理解。为了克服上述局限性,提出了概念聚类,使聚类的演化过程发生了质的变化。从概念聚类的角度来看,一些对象之所以聚成一类,一方面是因为它们在某些属性上彼此相似或是与其上下文存在某种内在的联系;另一方面是因为这些对象聚集在一起可以表达某一概念,而这一概念所表征的是这些对象作为一个整体所具有的共同性质。概念聚类由两个搜索过程组成:在概念层次空间的搜索,以确定较理想的概念层次结构;在可能的聚类空间的搜索,以确定较合适的划分以及在概念描述空间的搜索,为所产生的聚类赋予较合适的概念描述。但是由于搜索往往采用穷尽法或爬山法,因此存在着提高搜索效率或避免局部极小值等问题。

上述的聚类方法均没有考虑到聚类的目标,在一定程度上存在“答非所问”的现象。实际中普遍使用的是目标聚类法,其算法简单,应用范围广,并且可以转化为优化问题,借助于经典的非线性规划方法求解,便于计算机实现,因此目标聚类法是聚类研究的热点。

同时,聚类也是一个困难的问题。它运用数学方法研究和处理所给对象的分类以及各类别之间的亲疏程度,是在对数据不做任何假设的前提下进行分析。在人工智能和模式识别领域,聚类亦称为无先验学习或无监督学习。此外,现实世界中许多事物之间并无明显的划分,彼此之间的关系具有一定的模糊性和不确定性,需要将模糊集合理论引入聚类,即所谓的模糊聚类。

按照划分的结果,聚类可分为硬聚类和软聚类两种。硬聚类是指每个对象仅属于距离最近的聚类中心所属的类,即非此即彼。例如硬 c-均值(Hard C-Means,HCM)和 Kohonen

学习向量量化(Learning Vector Quantization,LVQ)等;软聚类是指每个对象以不同的隶属度或概率属于一个或多个类,如模糊 c-均值(Fuzzy C-Means,FCM)。显然,硬聚类是软聚类的一个特例。

下面将分别介绍硬聚类和模糊聚类。

6.1 硬聚类

6.1.1 概述

至今,聚类算法已有上百种之多,而且还有许多新方法在不断涌现。概括而言,聚类算法主要分为以下几类。

1. 基于划分的聚类算法

给定一个包含 n 个对象或元组的数据集合,构造数据的 k 个划分,每一个划分代表一个聚簇,并且 $k \leqslant n$。首先给定一个划分数目 k,划分方法是创建一个初始划分,然后利用迭代,通过对象在划分间移动以改进划分。为了达到全局最优,划分法可能穷举所有可能的划分。实际上,往往采用如下的启发式搜索方法,即:

(1) k 均值法。该算法中每一簇用该簇对象的平均值表示,使所有对象到聚类中心的距离平方和最小。

(2) k 中心点算法。该算法中每一簇用接近聚类中心的一个对象表示。

上述两种方法对于小的数据集合非常有效。为了处理大规模数据集,出现了 CLARA、CLARANS 等改进算法。此外,ISODATA 也是一种自动地进行类的合并和分裂的基于划分的聚类方法。

基于划分的聚类算法适用于数据集为凸形(见图 6.1)、类间距较远且直径相差不悬殊的情况,否则会出现错误。



图 6.1 凸集和非凸集

本质上,概念聚类是划分法的一种延伸,它用描述对象的一组概念取值将数据划分为不同的类,而不是基于几何距离实现数据对象之间的相似性度量。概念聚类能够输出不同类以确定其属性特征的覆盖,并对聚类结果进行解释。

2. 基于层次的聚类算法

基于层次的聚类算法是指对给定数据集进行层次分解。根据层次形成的方式,层次法分为凝聚(Agglomerative or Merging)和分裂(Divisive or Splitting)两种,如图 6.2 所示。

其中,凝聚法是指首先将每个对象作为单独的一个簇,然后相继合并相近的对象或簇,直到所有的簇合并为一个,或者满足终止条件;分裂法是指首先将所有对象置于一个簇中,

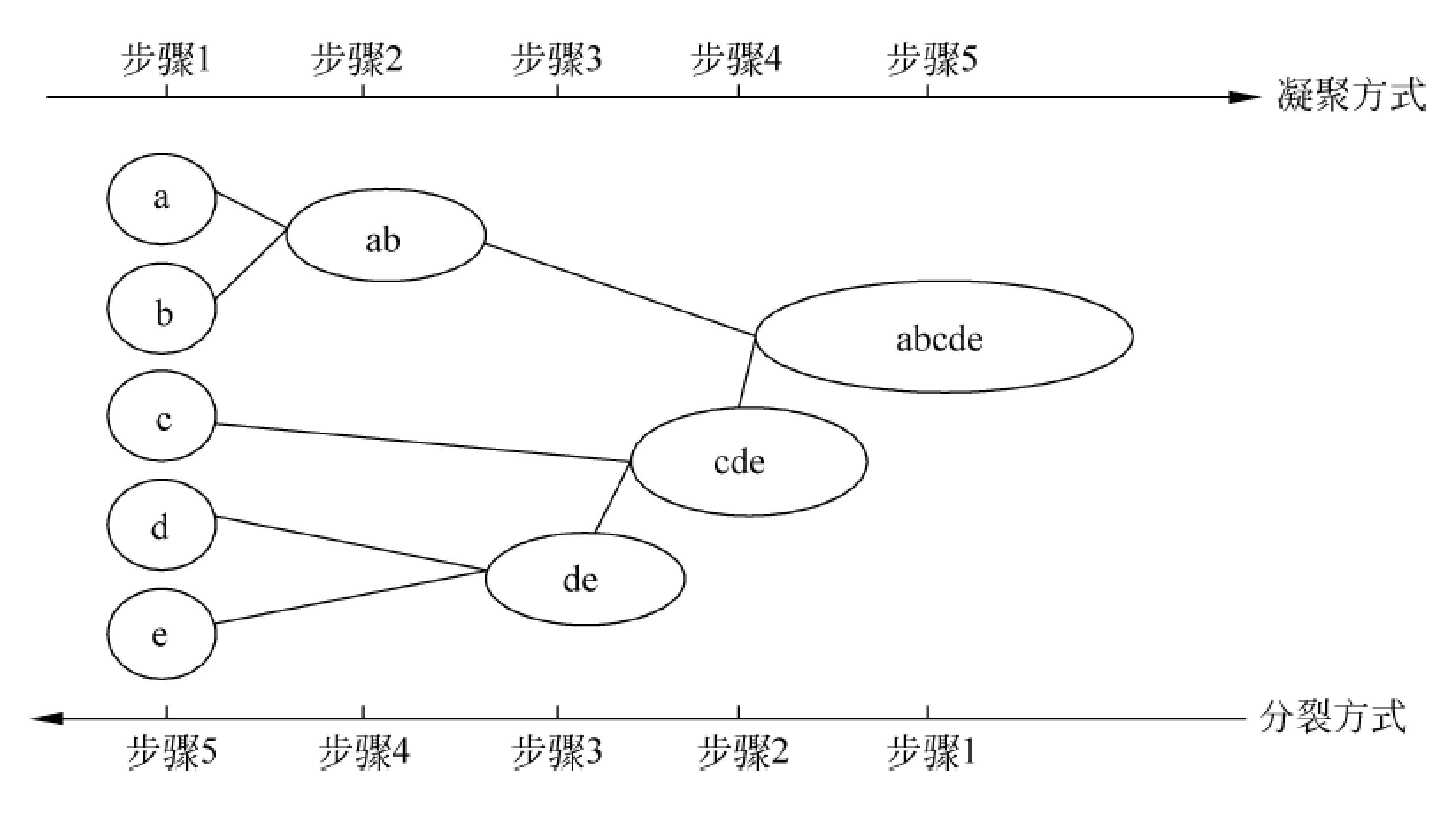


图 6.2　基于层次的聚类中的凝聚和分裂

通过迭代逐步将一个簇分裂为更小的簇。

根据不同的簇间距离度量方法，基于层次的聚类分为不同的种类，常用的度量方法包括最小距离、最大距离、平均值距离和平均距离等。

层次法无须参数，但需要定义终止条件。其缺点是一旦一个步骤（凝聚或分裂）完成，则不能被撤销。CURE、Chameleon 和 BIRCH 等均为改进的基于层次的聚类算法。其中，Chameleon 是一种凝聚的层次聚类算法，算法实现包括两个阶段，即首先将数据集划分为多个子集，然后将这些子集进行反复的合并，直至获得最终的聚类结果，如图 6.3 所示。

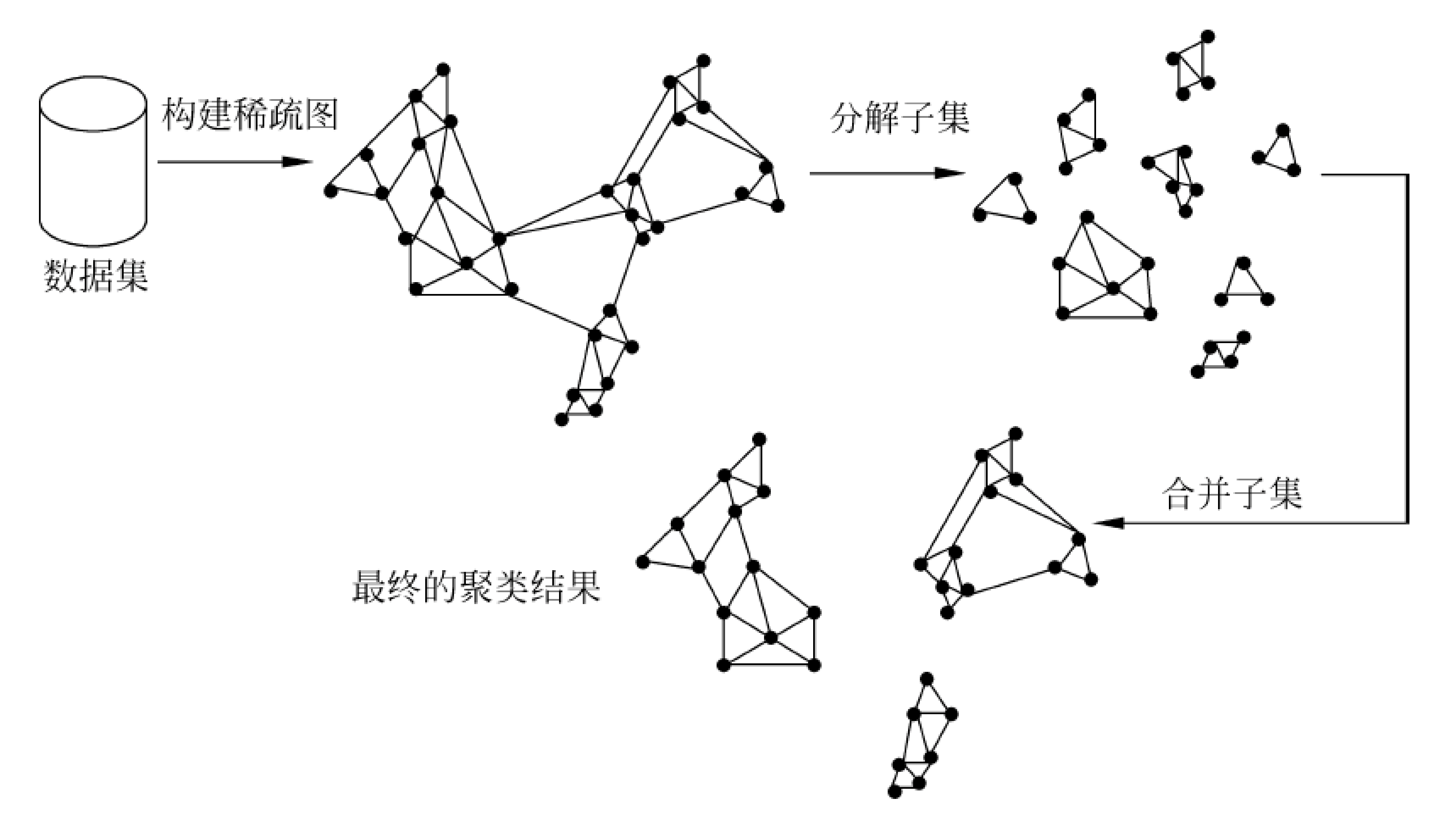


图 6.3　Chameleon 算法

CURE 算法如图 6.4 所示，其中 $s=50, p=2, s/p=25, q=5$。具体的算法步骤如下：

(1) 随机选取 s 个样本。

(2) 将所有样本划分为 p 个簇，每个簇样本数是 s/p。

(3) 将每个簇划分为 q 个子集，每个子集样本数是 s/pq。

(4) 删除孤立点数据。

- 随机取样。
- 如果一个簇变化缓慢,则删除该簇。

(5) 合并其中的部分子集。

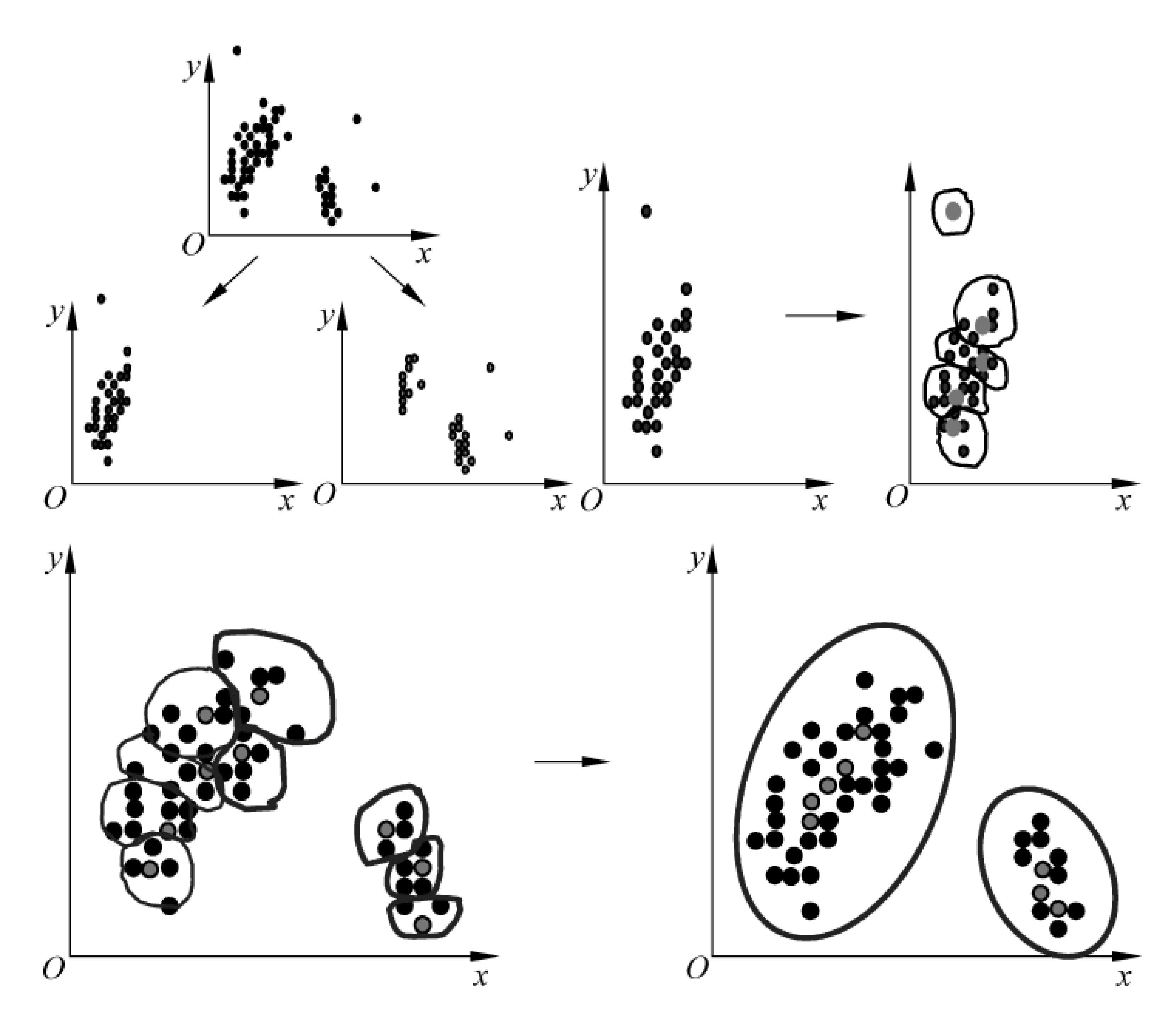


图 6.4　CURE 算法

基于层次的聚类算法适用于具有树枝状结构的数据集合。

3. 基于密度的聚类算法

这类算法的思想是只要某簇邻近区域的密度超过设定的阈值,就扩大簇的范围,继续聚类。这类算法可以得到任意形状的簇,典型算法包括 DBSCAN(Density-based Spatial Clustering of Applications with Noise)、OPTICS 和 DENCLUE 等。

4. 基于网格的聚类算法

基于网格的聚类算法首先将问题空间量化为有限数目的单元,形成一个空间网格结构,随后聚类在这些网格之间进行。其特点是聚类速度较快,典型算法包括 STING、WareCluster 和 CLIQUE 等。

5. 基于模型的聚类算法

基于模型的聚类算法是指为每个簇假定一个模型,寻找数据对给定模型的最佳拟合。典型算法包括 COBWEB 和神经网络算法等。

上述算法属于传统聚类的范畴。一般地,传统聚类算法对于维度较低的数据集有效,而当维度增加时,可能就不适用了。

针对大型数据库的聚类研究已提出了很多方法。例如,基于随机搜索的聚类方法 CLARANS、聚焦方法和聚类特征树法 BIRCH(平衡迭代消减聚类法)等。CLARANS 要求待聚类的对象必须事先调入内存,这对于大型数据库不太适用;聚焦方法通过引入 R 树,能够处理基于磁盘的大型数据库,但是 R 树的构造和维护代价太大;BIRCH 则是一种较为灵

活的递增式聚类方法，采用一个聚类特征三元组概括一簇对象的相关信息，从而以对应的聚类特征表示一簇对象而不是以具体的一组对象表示，通过构造满足分支因子和簇直径阈值的聚类特征树来实现聚类。BIRCH 算法需要提供适当的参数——聚类个数和簇直径阈值，这对于不具有可视化的高维数据是不可行的，而且对于一般的用户难以确定簇直径阈值。Agrawal 提出的 CLIQUE 算法，自顶向下求出各个子空间的聚类单元。CLIQUE 算法主要用于发现高维数据空间中存在的低维聚类，为了求出 k 维空间的聚类，必须组合出所有 $k-1$ 维子空间的聚类，导致其算法的空间和时间效率都较低，而且要求用户输入数据空间等间隔距离和密度阈值两个参数，这些参数与样本数据紧密相关，用户一般难以确定。BDCLASD 算法的基本思想是根据聚类中心的最近邻距离大于聚类外点的最近邻距离实现聚类。

6.1.2 相似度计算

在各种聚类算法中，数据对象之间特征差异通常是借助量化的指标进行表征，称为聚类统计量。聚类统计量包括距离或相似度。

一般地，数据对象采用向量表示，即通过一个在多维空间中的向量描述一个对象多方面的特征。向量的每个维度描述对象的一个特征，多个对象的向量构成一个模式矩阵(Pattern Matrix)，其中每行代表一个对象，每列描述一个特征，即 $(x_{ij})_{nm}$，其中 n 为对象数，m 为特征数，x_{ij} 为向量特征值。由于不同的特征采用不同的度量标准或尺度，这将对聚类结果产生影响，为此通常进行正规化，使所有的特征能用一个共同的标准度量。以下是一些常用的正规化方法：

(1) $x'_{ij}=\dfrac{x_{ij}}{\max|x_{ij}|}$ (6.1)

将所有的特征归一化到[−1,1]区间。

(2) 标准差正规化

$$x'_{ij}=\frac{x_{ij}-\mu_j}{\sigma_j} \tag{6.2}$$

其中，$\mu_j=\dfrac{1}{n}\sum_{i=1}^{n}x_{ij}$，$\sigma_j=\dfrac{1}{n}\sqrt{\sum_{i=1}^{n}(x_{ij}-\mu_j)^2}$，可使正态分布的特征取值主要集中在[−1,1]区间。

(3) $x'_{ij}=\dfrac{x_{ij}-\mu_j}{\sigma_j}$ (6.3)

其中，$\mu_j=\dfrac{1}{n}\sum_{i=1}^{n}x_{ij}$， $\sigma_j=\dfrac{1}{n}\sum_{i=1}^{n}|x_{ij}-\mu_j|$。

与式(6.2)相比，式(6.3)具有更大的适用范围，因此受噪声的干扰较小。

(4) 极差正规化

$$x'_{ij}=\frac{x_{ij}-\min\limits_{1\leqslant j\leqslant n}\{x_{ij}\}}{\max\limits_{1\leqslant j\leqslant n}\{x_{ij}\}-\min\limits_{1\leqslant j\leqslant n}\{x_{ij}\}} \tag{6.4}$$

将所有的特征归一化到[0,1]区间。

正规化后，最常用的对象间距离的计算公式包括：

(1) 欧氏距离,其定义为:

$$d_{ij}=\sqrt{|x_{i1}-x_{j1}|^2+|x_{i2}-x_{j2}|^2+\cdots+|x_{ip}-x_{jp}|^2} \tag{6.5}$$

其中,$i=(x_{i1},x_{i2},\cdots,x_{ip})$和$j=(x_{j1},x_{j2},\cdots,x_{jp})$分别是两个$p$维的数据对象。

(2) 曼哈顿距离(又称绝对距离),其定义为:

$$d_{ij}=|x_{i1}-x_{j1}|+|x_{i2}-x_{j2}|+\cdots+|x_{ip}-x_{jp}| \tag{6.6}$$

(3) 明考斯基(Mikulski)距离,是欧氏距离和曼哈顿距离的概化,其定义为:

$$d_{ij}=\left[\sum_{k=1}^{n}|x_{ik}-x_{jk}|^{\gamma}\right]^{1/\gamma} \tag{6.7}$$

其中$\gamma>0$。

n个对象彼此之间的相似度可通过相似度矩阵(Similarity Matrix)$(r_{ij})_{m}$表示,一个$n\times n$维、对角线元素为1的对称矩阵,其中r_{ij}是对象i和j之间相似度的量化表示,通常其值是非负的。对象i和j关系越亲密,其绝对值越接近于1;彼此关系越疏远,其值越接近于0。对象之间常用的相似度计算公式包括夹角余弦、相关系数和指数相似系数等。

(1) 夹角余弦

$$r_{ij}=\cos\alpha_{ij}=\frac{\sum_{k=1}^{n}x_{ik}x_{jk}}{\left(\sum_{k=1}^{m}x_{ik}^2\sum_{k=1}^{m}x_{jk}^2\right)^{1/2}} \tag{6.8}$$

其中,α_{ij}为向量$(x_{i1},x_{i2},\cdots x_{im})$和$(x_{j1},x_{j2},\cdots x_{jm})$之间的夹角。

(2) 相关系数

$$r_{ij}=\frac{\sum_{k=1}^{m}|x_{ik}-\bar{x}_i||x_{jk}-\bar{x}_j|}{\left[\sum_{k=1}^{m}(x_{kj}-\bar{x}_i)^2\sum_{k=1}^{m}(x_{kj}-\bar{x}_j)^2\right]^{1/2}} \tag{6.9}$$

(3) 指数相似系数

$$r_{ij}=1/m\sum_{k=1}^{m}\exp\left[-\frac{3}{4}\frac{(x_{ik}-x_{jk})^2}{S_k^2}\right] \tag{6.10}$$

为了实现自动聚类,首先定义相似度度量,无论采用哪种度量方法有一点是明确的,即相似度的度量与具体问题有关,给出通用的度量是十分困难的。例如,采用欧氏距离度量相似度的聚类是将聚类模式看作若干个超球体的集合,适合于数据特征空间呈超球体的聚类。实际上,这一度量方法对于沿主轴分布的情况则是无用的甚至是不合理的。又如,实数集合的聚类结构表现为超椭球体,因此对于样本分布呈超椭球体的情况,常常采用Mahalanobis距离,即:

$$D(\boldsymbol{x},\boldsymbol{m})=(\boldsymbol{x}-\boldsymbol{m})^{\mathrm{T}}\sum_{x-m}^{-1} \tag{6.11}$$

其中,$\sum$是该聚类群体的协方差矩阵,$\boldsymbol{m}$为平均向量,$\boldsymbol{x}$代表该聚类。

采用Mahalanobis距离的主要困难在于:每当改变某聚类范围时,都需要计算一次样本协方差矩阵的逆。以Mahalanobis距离为相似度度量的自组织神经网络实现分布呈超椭球体的聚类(Hyper-Ellipsoidal Clustering,HEC)可减少计算量。HEC分为两层:第一层由大量的主成分分析子网组成,以判断当前已形成的分布为超椭球体的聚类;第二层则是利用第一层所提供的聚类信息进行竞争学习。该方法不需要计算样本协方差矩阵的逆。但

是，若学习参数选择不当，则实现主成分分析的子网达到收敛需要较长的时间。

6.1.3 实现方法

聚类的实现方法可概括为三种。

1. 基于目标函数迭代的实现

在优化目标函数的过程中，人们曾经尝试过动态规划、分支定界和凸切割等方法，然而大量的存储空间和运行时间限制了其应用。实际应用最为广泛的是 Dunn 提出的迭代优化算法——k-均值。本质上，迭代优化属于局部搜索的"爬山法"，易陷入局部极值，对初值也较敏感。

假设将 n 个样本 $x_j \in \Re^N (j=1,2,\cdots,n)$ 划分为 c 类，对 $i=1,2,\cdots,c$ 和 $j=1,2,\cdots,n$ 可以定义：

$$\mu_{ij} = \begin{cases} 1, \text{如果第 } j \text{ 个样本属于第 } i \text{ 类} \\ 0, \text{否则} \end{cases} \tag{6.12}$$

则矩阵 $\boldsymbol{\mu}=(\mu_{ij})$ 具有如下性质：

$$\mu_{ij} \in \{0,1\} \text{且} \sum_{i=1}^{c} \mu_{ij} = 1 \qquad (j = 1,2,\cdots,n) \tag{6.13}$$

设 n_i 表示第 i 类中所包含的样本个数，则

$$n_i = \sum_{j=1}^{n} \mu_{ij} \quad i = 1,2,\cdots,c \tag{6.14}$$

设 $\bar{x}_i \in \Re^N$ 是第 i 类的中心，则

$$\bar{x}_i = \frac{\sum_{j=1}^{n} \mu_{ij} x_j}{\sum_{j=1}^{n} \mu_{ij}} = \frac{1}{n} \sum_{j=1}^{n} \mu_{ij} x_j \qquad i = 1,2,\cdots,c \tag{6.15}$$

故第 i 类的类内差为：

$$S^{(i)}(\mu) = \sum_{j=1}^{n} \mu_{ij} \, \| x_j - \bar{x}_j \|^2 \tag{6.16}$$

整体类内差为：

$$S(\mu) = \sum_{i=1}^{c} S^{(i)}(\mu) = \sum_{i=1}^{c} \sum_{j=1}^{n} \mu_{ij} \, \| x_i - \bar{x}_i \|^2 \tag{6.17}$$

公式(6.17)为经典的类内平方误差和(Within-Group Sum of Squared Error，WGSS)目标函数。k-均值旨在通过迭代优化寻找 μ_{ij}^*，使得 $S(\mu)$ 取最小值，即：

$$S(\mu^*) = \min\{S(\mu)\} \tag{6.18}$$

由公式(6.18)可知，当各样本独自成为一类时，即 $c=n$，$S(\mu)$ 取最小值 0。因此仅仅凭借该目标函数是无法找到最优分类的，必须考虑其他条件，即寻找一个合适的目标函数。k-均值是在给定 c 的前提下，优化 $S(\mu)$。因此，对聚类个数 c 而言，本质上是一种枚举法。

2. 基于神经网络的实现

采用神经网络实现聚类的显著优势在于神经网络的并行处理。因为在数据量庞大的情况下进行聚类是相当耗时的。

Kohonen 学习向量量化和自组织特征映射在经典模式识别领域中的一个重要应用就是聚类。近年来,Kohonen 受到普遍关注,然而 Kohonen 聚类网络(Kohonen Clustering Network,KCN)启发式的演化过程使其在应用中存在着网络收敛依赖于输入样本顺序以及难以保证收敛等若干致命的问题。一些学者基于不同的角度和背景,提出了改进算法,但仍存在某些缺陷。后来,提出了一种用于超椭球体聚类的神经网络,它能自适应地估计每一类超椭球体形状,并将所得到的信息用于竞争学习,通过引入正规化的 Mahalanobis 距离,防止过大或过小类的产生。此外还提出了一种带惩罚项的竞争学习算法。

3. 基于进化计算的实现

进化计算是建立在生物进化基础上的基于自然选择和群体遗传机制的随机搜索算法。由于具有全局并行搜索的特点,因此可以较高的概率获得全局最优解。此外,进化计算还具有简单、通用和鲁棒性等优势。所以,人们将进化计算引入聚类,形成了一系列基于进化计算的聚类算法。它们大致可分为两类:一是基于模拟退火(Simulated Annealing,SA)的方法;二是基于遗传算法和进化策略的方法。

表 6.1 列出了三种聚类实现方法的比较。

表 6.1 三种聚类实现方法的比较

技术指标	基于目标函数迭代的实现	基于神经网络的实现	基于进化计算的实现
搜索方法	梯度下降法	梯度下降法	随机搜索法
收敛速度	较快	快	慢
算法精度	高	较高	受编码长度的限制
算法结构	串行	并行	并行
初值敏感度	敏感	敏感	不敏感

6.1.4 主要算法

下面介绍主要的硬聚类算法。

1. 基于目标函数的聚类

基于目标函数的聚类主要包括 k-均值、k-中心点和 c-均值等算法。

1) k-均值

k-均值也称为硬 c-均值,该算法首先由 Mac Queen 提出,是一种基于划分而非分层的聚类方法。k-均值的基本思想是:首先随机地选择 k 个对象,每个对象初始地代表一个簇的平均值或中心。对于其余的每个对象,根据其与簇中心的距离,将其划分到最近的簇,然后重新计算每个簇的平均值。此过程不断迭代,直到准则函数(或称为相似度函数)收敛,即公式(6.19)中的函数值最小:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2 \tag{6.19}$$

公式(6.19)中,E 是所有对象的平方误差和,m_i 是簇 C_i 的平均值,p 是空间中的点(p 和 m_i 都是多维的),此准则函数试图使生成的簇尽可能地独立和紧凑。k-均值是将 n 个对象划分为 k 个簇,使簇内对象的相似度较高,而簇间对象的相似度较低。

k-均值聚类算法的步骤如下:

输入：聚类数目 k，包含 n 个样本的数据集合

输出：各样本属于 k 个簇的信息

(1) 随机选择 k 个样本作为初始的簇中心；

(2) 将剩余的 $n-k$ 个样本按照与簇中心的距离划分到最近的簇；

(3) Repeat；

(4) 计算各个簇中样本各属性的平均值，作为新的簇中心；

(5) 重新将 n 个样本按照与簇中心的距离划分到最近的簇；

(6) Until　簇中心不再变化。

---

k-均值是一种经典算法，其主要优点是算法简单、快速而且能有效地处理大型数据。但是此算法对不同的初始值可能会导致不同的聚类结果，执行结果与输入顺序有关。其次，这种算法易陷入局部极小值。这两大缺陷大大限制了其应用范围。

2) k-中心点

k-中心点是对 k-均值的改进。不采用簇中对象的平均值作为参照点，而是选用簇中位置最中心的对象，即中心点。这样的划分方法依然是基于最小化所有对象与其参照点之间的相异度之和的原则进行的。

k-中心的基本思想是：首先为每个簇随机地选取一个数据样本作为中心点，将剩余的数据样本依照距离的远近分配给最近的簇；随后选取其他的非中心点数据做中心点，并查看聚类情况。如果替换的聚类总代价小于零，则执行替换直到中心点不再发生变化，即达到代价最小值时算法停止。

k-中心点聚类算法的步骤如下：

输入：聚类数目 k，包含 n 个样本的数据集合

输出：各样本属于 k 个簇的信息

(1) 随机选择 k 个样本作为初始的簇中心点；

(2) Repeat；

(3) 将非中心点的样本依照与各簇中心点的距离划分到最近的簇；

(4) 随机地在非中心点中选择一个样本；

(5) 计算使用该点做中心点来代替原中心点的代价 S；

(6) If $S<0$ Then 用该点替换原中心点，形成新的簇；

(7) Until　簇中心点不再发生变化。

---

3) c-均值

c-均值聚类算法的步骤如下：

输入：聚类数目 k，包含 n 个样本的数据集合

输出：各样本属于 k 个簇的信息

(1) 随机选择 k 个样本作为初始的聚类中心 $c_1, c_2, \cdots, c_k$；

(2) 将每个样本向量 $x_l=[x_{l1},x_{l2},\cdots,x_{ln}]$，其中 n 为向量的维数，按欧氏距离 $\|x_l-c_i\|=\min\limits_j\|x_l-c_j\|$ 归入聚类中心为 c_i 的类；

(3) 重新调整聚类中心 c_i，令 $c_i=[c_{i1},c_{i2},\cdots,c_{in}]$，其中 $c_{im}=\dfrac{\sum\limits_{x_{li}\in \text{cluster}i} x_{l_i m}}{N_i}$，$N_i$ 是第 i 个类别中的向量数；

(4) 如果步骤(3)中的聚类中心不再变化，则算法停止；否则，转至步骤(2)。

---

2. 基于神经网络的聚类

基于神经网络的聚类主要包括竞争学习(Competitive Learning)和自组织映射(Self Organizing Maps，SOM)等方法，都涉及竞争的神经元。

1) 竞争学习

前馈神经网络的竞争式学习规则是由 Rumelhart 和 Zipser 提出的，由若干个单元(神经元)组成层次结构，以一种“胜者全取”实现竞争。竞争学习中各层之间的联接是激发式的，即在某个给定层次的单元可以接受来自低一层次所有单元的输入，在一层中活动单元的布局代表了高一层的输入模式。在某个给定层次中，一个簇中的单元彼此竞争，对低一层的输出模式做出反应。任何簇中只有一个单元是活跃的。获胜的单元修正其与簇中其他单元的连接权重，以便在未来能够对与当前对象相同或相似的对象做出较强的反应。如果将权重看作一个标本，那么新的对象将被划分到具有最近标本的簇。

聚类结束后，每个簇被认为是一个新的“特征”，代表对象的某些规律，因此产生的簇可以看作一个低层特性向高层特性的映射。

2) SOM

SOM 是神经网络最重要的模型之一，是 1982 年 Teuvo Kohonen 基于对生物神经活跃区域的模拟提出的。它是一种无监督的聚类方法，通过反复学习和若干个单元的竞争实现聚类。权重向量最接近当前对象的单元成为活跃或获胜的单元。为了更接近输入对象，对获胜单元及其最近邻的权重进行调整。SOM 假设在输入对象中存在一些拓扑结构或顺序，单元将最终在空间呈现这种结构。单元的组织形成一个特性映射。SOM 被认为类似于大脑的处理过程，对在二维或三维空间中可视化高维数据是很有用的。

SOM 由输入层、竞争层和输出层组成。输入层结点的数目同输入样本的特征向量维度相同，输出层的每个结点都是一个含有同输入层结点个数相同维数的向量。SOM 网络是全连接的，每个输入结点都与所有的输出结点连接。

假定输入向量维度为 N，输出结点数为 M。SOM 算法的描述如下：

(1) 建立一个有 M 个输出结点的二维网格，初始化从 N 个输入层结点到 M 个输出层结点的权值 w_{ij} 为[0,1]之间的随机数。

(2) 输入 N 维的样本向量。

(3) 计算输入向量在时刻 t 到所有输出结点的距离，即

$$d_j=\sum_{i=0}^{N-1}(x_i(t)-w_{ij}(t))^2 \tag{6.20}$$

其中，w_{ij} 为输出结点的权值，$x_i(t)$ 是输入向量在时刻 t 的值。

(4) 选择获胜结点并更新它和邻近结点的权值。

选择产生最小 d_j 的结点为获胜结点 j，并更新其及邻近结点的权值以减小到输入向量 $x_i(t)$ 的距离，即：

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(t)(x_i(t) - w_{ij}(t)) \tag{6.21}$$

每次更新后，j 及其邻近结点都会更接近输入向量 $x_i(t)$。$\alpha(t)$ 是误差调整系数，$0<\alpha(t)<1$，并且随时间 t 单调减小，w_{ij} 的调整也越来越小，直至收敛。

SOM 以其无监督学习、可视化等特性，广泛应用于聚类分析、图像处理、语音识别和组合优化等众多领域。然而，传统的 SOM 也存在着许多不足，其最大局限性是在学习样本量较少时，网络连接权重的初始值对收敛影响很大，而且聚类效果取决于样本的输入顺序。

3. 基于进化计算的聚类

基于进化计算的聚类大致可分为两类：一是基于模拟退火(Simulated Annealing，SA)的方法。确定性退火技术是美国的 K. Rose 博士于 1990 年首先提出的，已得到了一些比较满意的理论结果。后来提出了一种利用确定性退火的启发式聚类算法，把聚类问题看作是一个物理系统，通过求解一系列随温度变化的自由能量函数的全局极小获得聚类的最优解。但是，模拟退火算法只当温度下降得足够慢时才能收敛到全局最优，大量的运算时间限制了其实用性；二是基于遗传算法和进化策略的方法。

常用的进化算法和策略包括模拟退火、遗传算法、蚁群算法和粒子群优化算法等。

1) 模拟退火

SA 算法的出发点是物理中固态物质的退火过程与一般组合优化问题之间的相似性。固态物质退火时，通常先将之加温，使其中的粒子能够自由移动，然后逐渐降低温度，粒子也逐渐形成低能态的晶格。若在凝结点附近温度的下降速度足够慢，则固态物质一定会形成最低能量的基态。

SA 算法中固体状态对应组合优化问题的可行解，最低能量的基态对应最优解，逐渐降低温度的过程对应控制参数的下降。SA 首先由某一较高初始温度开始，伴随温度参数的不断下降重复抽样，最终得到问题的全局最优解。SA 包括一个温度持续下降的过程，能够避免局部最小，是一个基于概率的全局最优启发式方法。

在温度 T 时，由当前状态 i 产生新状态 j，两者的能量分别为 E_i 和 E_j。若 $E_j<E_i$，则接受新状态 j 为当前状态；否则，若概率 $\exp\left(-\frac{E_j-E_i}{kT}\right)$ 大于[0,1)区间内的随机数，则仍旧接受新状态 j 为当前状态；若不成立，则保留状态 i 为当前状态。$\exp\left(-\frac{E_j-E_i}{kT}\right)$ 中 k 为 Boltzmann 常量。这种方法使得能量为 E_i 的状态成为当前状态的概率是：

$$\frac{\exp(-E_i/kT)}{\sum_j \exp(-E_j/kT)} \tag{6.22}$$

这一概率函数称为 Boltzmann 浓度，其特点是对于较高的温度，每一状态都具有相同的概率成为当前状态，而对于较低的温度，仅仅那些低能量的状态才具有较高的概率成为当前状态。

标准 SA 算法的描述如下：

(1) 随机产生一个初始状态 S_0，$S_i = S_0$，令 $k=0$，$T_0=T_{max}$(初始温度)。

(2) 若在该温度达到内循环停止条件,则转步骤(3);否则,从邻域 $N(x_i)$ 中随机选一状态 S_j,若 $\Delta=E_j-E_i<0$,则 $S_i=S_j$;否则若 $\exp\left(-\frac{\Delta}{T_k}\right)>\text{random}[0,1]$ 时,则 $S_i=S_j$;重复步骤(2);

(3) 退温 $T_{k+1}=d(T_k)$,$k=k+1$;若满足终止条件,停止计算;否则转步骤(2)。

通常设 $T_{\max}=100$,步骤(3)中的退温函数 $d(\cdot)$ 可采用

$$T_{k+1}=\frac{T_k}{1+\tau\sqrt{T_k}} \tag{6.23}$$

其中 τ 为一个小的时间常数。

SA 算法的优点是:

- 通用性强,能够处理任何系统和费用函数,即使对复杂问题 SA 的编码也相对容易。
- 通常可保证找到问题的全局最优解。普通的梯度下降算法总是向改进解的方向搜索,这种“贪心”算法往往导致只能找到一个局部最优解,而不是全局最优解。如图 6.5 所示,SA 算法中,在系统能量减少这样一个总的趋势下,允许搜索偶尔向能量增加的方向搜索,以避开局部极小,最终能够稳定到全局最优状态。

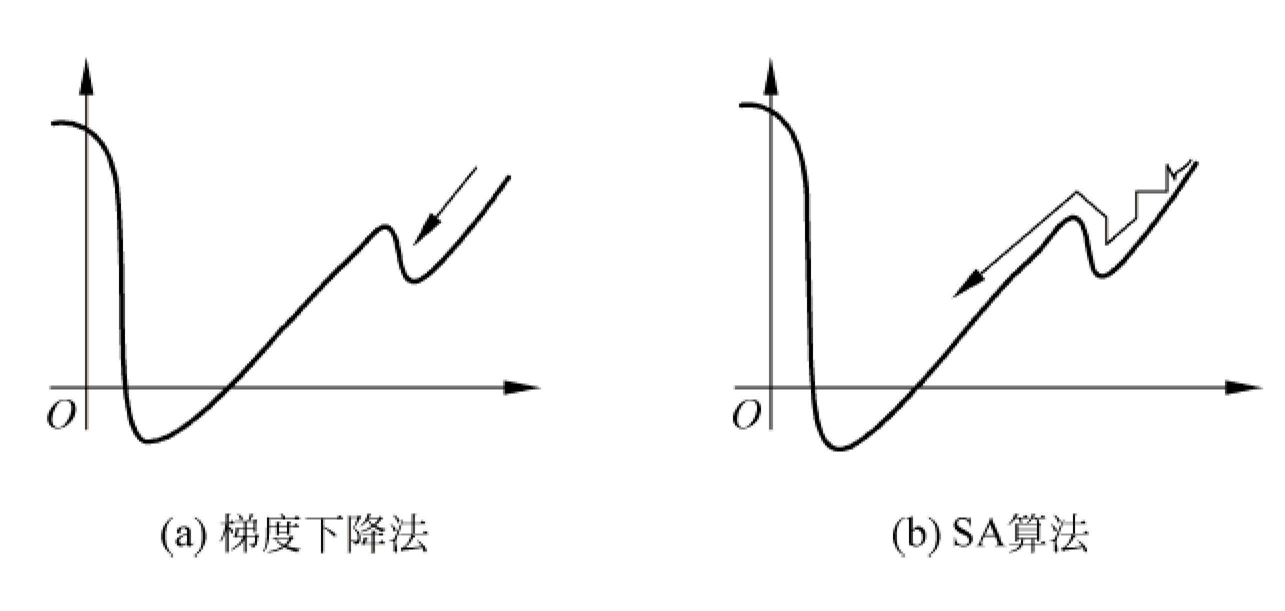


图 6.5　梯度下降法与 SA 算法搜索过程的对比

SA 为寻找到最优解,算法通常要求较高的初始温度、较慢的降温速率、较低的终止温度以及各温度下足够多的抽样,因而 SA 算法往往优化过程较长,这是 SA 算法的不足之处。因此,在保证一定优化质量的前提下提高算法的搜索效率,是改进 SA 算法的主要途径。

2) 遗传算法

遗传算法(Genetic Algorithm,GA)由 Holland 在 1975 年提出,更早的思想可以追溯到 20 世纪 60 年代。典型的 GA 使用独立于问题的表示形式,即二进制位串,这种编码既适合于变异,又适合于交叉,并且强调交叉算子的搜索能力。随后,Holland 将 GA 引入到自适应系统,后来又推广到其他领域。

GA 的进化对象是由多个个体(individuals)组成的种群(population)。在初始化后,通过基于适应度的概率选择算法选择父代,并通过交叉(crossover)和变异(mutation)维持种群的多样性。如此演化下去,直到满足终止条件。

基于自然界中生物遗传与进化机制的模仿,针对不同的问题,设计了多种不同的编码方法来表示问题的解空间,设计了各种遗传算子来模仿不同环境下的生物遗传特性。这样,就由不同的编码方法和遗传算子构成了各种遗传算法。但它们都具有共同的特点,即通过模仿生物遗传和进化过程中的选择、交叉和变异等,完成对问题最优解的自适应搜索过程。基

于这一共同特点，Goldberg 总结出了一种最基本的遗传算法——基本遗传算法（Simple Genetic Algorithm，SGA）或 CGA（Canonical GA）。SGA 只使用选择、交叉和变异三种算子，其遗传操作过程简单、容易理解，是其他一些遗传算法的基础和雏形，不仅给各种改进的遗传算法提供了一个基本框架，同时也具有一定的应用价值。

综上所述，遗传算法和其他进化算法相比的显著特征是使用交叉算子产生下一代个体；使用繁殖选择式的选择算子、根据适应度随机选取父本；一般使用二进制位串编码及相应的交叉算子。

SGA 算法描述如下：

```
随机化初始种群 P(0),t = 0.
While(不满足终止条件) do
{
  计算所有个体的适应度
  计算每个个体的选择概率
  均匀随机选择 pg * N 个个体,直接插入到下一代种群 P(t + 1)中.
  for(i = 0; i<(1 - pg) * N/2; i++)
     {
       根据选择概率在 P(t)中选择两个父本
       r = random(0,1).
       If r < Pc,将两个父本不加改变地插入到下一代种群 P(t + 1)中
       Else
         {
           执行重组操作,产生两个子代.
           按照变异概率 pm 对两个子代执行变异操作.
           将其插入到 P(t + 1)中
         }
     }
}
```

上述算法中包含四个基本参数，分别是代间隔 p_g、交叉概率 p_c、变异概率 p_m 和种群规模 N。

概括起来，SGA 的要素包括：

（1）染色体编码方法

编码（Coding）是将问题空间中的点映射到基因空间的过程。基因空间既可以由二进制位串表示，也可以是实空间 R^n 的一个子集。当基因空间的编码方式为位串时，也称为染色体（Chromosome），其中的每一位称为基因（Gene）。基因的取值范围叫做等位基因（Allele）。

编码方法决定了基因型和表现型之间的转换方法，某些特定类型的编码方法还决定了遗传算子的选择。编码的好坏在很大程度上决定了算法的优劣。

编码方案取决于具体的问题，因此目前尚没有一定的理论和评价原则。作为参考，De Jong 提出了两条操作性比较强的编码原则：

- 积木块原则：应使用能易于产生与所求问题相关的具有低阶、短定义长度的编码方案。
- 最小字符集编码原则：应使用能使问题得到自然表示或描述的具有最小编码字符集的编码方案。

这两条原则具有一般的指导意义。随着时间的推移,使用的编码方式也越来越丰富。在实际使用时,还需要考虑其他的原则。尤其是对于约束优化问题,一种方法是在编码时避免不合法个体的存在,这样有可能造成编码和算子的复杂化与不一致性;另一种方法是针对不合法的个体,对其适应度加以惩罚。

① 位串编码

在进行编码时,将基因看作一个有序的位串序列,而不考虑每一位的含义。这种编码类似于生物染色体的组成,使交叉和变异等遗传操作很容易进行。

使用位串编码时,二进制编码可同时表示的模式数最多。并且实现简单,对于很多离散优化问题(如背包问题等),基因型与表现型的对应关系非常明确。因此得到广泛应用。

对于连续优化问题,二进制编码的主要问题是:相邻整数之间的 Hamming 距离可能很大,带来很多不必要的局部极值点,影响算法的搜索性能。这一问题也称为 Hamming 悬崖(Hamming Cliff)。

克服这一缺点的一种方法是使用 Gray 编码。Gray 编码与二进制编码的对应关系如下:

设二进制串$(\beta_1,\beta_2,\cdots,\beta_{n-1},\beta_n)$,对应的 Gray 编码串为$(\gamma_0,\gamma_1,\cdots,\gamma_{n-1},\gamma_n)$,则:

$$\gamma_k=\begin{cases}\beta_1,\text{若 } k=1\\ \beta_{k-1}\oplus\beta_k,\text{其他}\end{cases} \tag{6.24}$$

同样

$$\beta_k=\begin{cases}\gamma_1,\text{若 } k=1\\ \gamma_{k-1}\oplus\gamma_k,\text{其他}\end{cases} \tag{6.25}$$

这里$\oplus$表示模 2 加法,即异或操作。

Gray 编码的显著特点是对于距离为 1 的二进制表示,其 Gray 编码之间的 Hamming 距离为 1。因此可以在一定程度上克服上述缺点。

② 实数编码

当问题空间是实数连续空间时,可以直接采用实数进行编码。对于实数编码,从理论上讲,二进制编码的各种遗传操作都可以使用,但实际应用时通常都使用专门针对实数编码设计的算子。从进化计算的历史来看,进化策略和遗传规划都采用实数编码。近年来,遗传算法在求解复杂连续优化问题时也经常使用实数编码。实际上,使用实数编码的遗传算法和进化策略的区别已经越来越小。

③ 结构化编码

对于很多具有明确的数据结构的问题,更加自然地表示是直接对这种数据结构进行操作,称之为结构化编码。常见的编码方式有树和图。这种编码方式一般是针对具体问题设计具体的编码和遗传算子,很难具有通用性。对于由 Koza 提出的遗传规划(Genetic Programming),可以看作是使用逆波兰表达式的二叉树作为结构化编码的进化算法的例子。

对于 SGA 最常用的是位串编码,即使用固定长度的二进制符号串表示种群中的个体,其等位基因是由二进制符号集$\{0,1\}$组成的。随机产生 N 个初始字符串,每个字符串称为一个个体(Individual),N 个个体构成一个初始种群。SGA 以 N 个个体为起点开始迭代。

(2) 适应度函数

对于二进制位串空间 $\Omega=[0,1]^l$，l 称为染色体长度。称问题空间中的点为表现型(Phenotype)，基因空间中的点为基因型(Genotype)。对于一个特定的基因型，其对应的表现型的优化函数值称为适应度(Fitness)。

适应度函数表明个体对环境适应能力的强弱，是自然选择的唯一参考因素。

当欲求解的原始问题是数值优化问题时，可以直接将求解函数作为适应度。个体的适应度取值通常为正的实数值。一般情况下，当个体的性能越好时，其适应度值越大，而且要求非负(如 GA 中的比例选择策略)。因此，有时需要对原始的适应度函数进行变换。

当原始问题是非数值优化问题时，一种方案是选择恰当的度量函数充当适应度函数，将某个可行解的适应度变换到正实数空间；另一种方案是使用不基于适应度函数具体数值的选择策略。如排名选择和锦标赛选择。

在很多情况下，原始的适应度函数(及其简单变换)存在一些不适合选择使用的特点。有时，适应度值之间的差别较小，导致选择效果不明显；有时，优势个体的适应度值过大，可能产生早熟收敛。在这些情况下，需要对原始的适应度函数进行某种变换，以得到更好的性能。这种变换的具体形式通常是通过经验和试验获得的。

SGA 算法按与个体适应度成正比的概率确定当前种群中每个个体遗传到下一代种群的机会。

(3) 遗传算子

一般而言，各种进化算法的不同点在于产生新个体与选择的方式不同。这种方式也称为算子(Operator)。进化算子可以分为两类：选择算子和演化算子。在有些算法的具体实现中，这两种算子是混合在一起的。

选择算子充当自然进化中自然选择的角色，起到指引搜索方向的作用。其目的是提高具有较高适应度的个体或其后代存活的概率。通过选择算子，可以使种群向更高适应度的方向前进。不同的选择算子导致不同的选择压力(Selection Intensity)。选择压力较大，算法的收敛速度较快，但也容易导致早熟收敛。

选择算子按照选择阶段可分为繁殖选择和生存选择；按照比较范围可分为种群选择和生境选择；按照计算方式可分为确定性选择和概率性选择。繁殖选择指通过选择确定哪些个体可以用来产生下一代；生存选择指通过选择确定哪些个体可以存活。种群选择指选择是在整个种群的范围内进行的，个体的适应度要和整个种群的适应度分布进行比较；生境选择指选择是在两个或几个个体(通常具有血缘关系)之间进行的。确定性选择使用确定性的算法进行选择；而概率性选择在选择过程中引入了随机性的因素。

演化算子充当自然进化中繁殖过程中遗传和变异的角色，起到维护种群个体构成多样性(Diversity)的作用。它包括交叉(Crossover)和变异(Mutation)两种。交叉算子又称重组(Recombination)算子，用于从两个父本产生一个新的个体。演化算子在进化算法中起到构造的作用，可以从一个、两个或多个个体出发，构造出新的个体。其目的是产生和维护种群的多样性，同时起到局部搜索的作用。

通常 SGA 使用以下三种遗传算子：

① 选择算子(Selection Operator)

一个种群中同时有 N 个个体存在，这些个体哪个保留用于繁殖后代，哪个被淘汰，是通

过选择过程实现的。进行选择的原则是适应度大的个体为下一代贡献一个或多个后代的概率较大。选择体现了“优胜劣汰”的原则。

常用的选择策略包括：

- 基于适应度比例的选择
- 基于排名的选择(Ranking Selection)

在使用基于适应度比例的选择策略时,会出现由于某些个体的适应度过大而导致的早熟收敛。同时这种方式依赖于适应度函数的具体形式,对于某些应用而言,影响计算的收敛性和收敛速度。

基于排名的选择策略根据个体在整个种群中的适应度的好坏排名来分配选择概率,可以避免上述问题。

这种算法的基本形式是先根据所有个体的适应度进行排序,设排序后的次序为 1..N。然后对于每一个个体指定一个选择概率函数 p_i。p_i 只与 i 有关,与 f_i 无关;且满足 $\sum_{i=1}^{N} p_i = 1$,以及对于 i 是单调递减的。

常见的选择概率函数的形式有线性排名和指数排名两种。

对于线性排名
$$p_i = \frac{1}{N}\left(a - \frac{2*(a-1)*i}{N+1}\right) \tag{6.26}$$

通常取 $a=1.1$

对于指数排名
$$p_i = \begin{cases} q(1-q)^{i-1}, i<N \\ (1-q)^{N-1}, i=N \end{cases} \tag{6.27}$$

基于排名的选择策略可以看作是对适应度函数的一种自适应变换,只不过这种变换是动态的,随着种群进化而变化,同时只与适应度的相对值有关,而与其绝对值无关。

- 基于局部竞争的选择

以上两种方案都是根据个体在整个种群中的相对地位决定其选择概率,因此需要整个种群的信息。当种群规模很大时,需要一定的额外计算量。同时也不利于并行计算。基于局部竞争的选择策略可以在一定程度上解决这个问题。

锦标赛选择(Tournament Selection)通过在随机选出的若干个个体之间进行竞争,适应度最大的个体获得优胜,从而被选出产生其后代。

② 交叉算子(Crossover operator)

对于选中的用于繁殖的个体,随机选择位置,交换字符串左边部分,产生新个体,新个体继承了其父辈的特性。交叉体现了信息交换的思想,它是算法的核心。

交叉算子的具体形式和基因的编码方式密切相关。对于位串编码,最简单的交叉算子是单点交叉,也有复杂一些的多点交叉和均匀交叉算子。

单点交叉的实现如下:随机地在两个父本上选择一个交叉点,然后交换这两个串对应的子串,得到的子代分别由父本的连续的一部分构成。多点交叉则是随机生成多个交叉点,然后间断交换父本中对应的子串。均匀交叉则是依概率交换父串中的每一位。

对于这三种交叉算子,可以使用统一的形式化描述:

设两个父本为 j_1 和 j_2,存在一个交叉模板 m,交叉后的子代分别为:

$$s_1 = (j_1 \otimes m) \oplus (j_2 \otimes \bar{m}),\ s_2 = (j_1 \otimes \bar{m}) \oplus (j_2 \otimes m) \tag{6.28}$$

在公式(6.28)中,⊗、⊕和$\bar{m}$分别表示二进制的与、异或和补运算。

如果 m 的结构为在某位之前都为 1,其后都为 0(或相反),则公式(6.28)表示单点交叉;若 m 由连续的 0 串和 1 串混合而成,则表示多点交叉;而 m 的一般形式代表均匀交叉。

从模式的角度来看,多点交叉和均匀交叉能够搜索到的模式更多,具有更强的搜索能力。但实践证明,多点交叉和均匀交叉的作用并不显著。

对于其他类型的编码方式,可以设计出不同的交叉算子。在实际使用时,为避免致死基因(lethal gene),也需要对交叉算子进行改进。实际上,很多特定领域中使用的 GA 都要为之设计包含领域知识的交叉算子。

③ 变异算子(Mutation operator)

变异操作是把一个个体的每一位按照概率取反。同生物界一样,GA 中发生变异的概率很低,通常取值在 0.001~0.01 之间。

在二进制编码中,基本的变异方式就是将某一位或某几位进行反转。其形式化的描述是:

设父代的二进制编码为 j,存在一个变异模板 m,变异后的结果为:

$$U(j) = j \oplus m \tag{6.29}$$

具体地,存在两种典型的变异方式:1—比特变异和 c/l 变异。

1—比特变异指按照变异概率确定待变异的个体后,均匀地从中选取一位进行反转。也就是模板中只有一位是 1;c/l 变异指对于某个个体,每一位都按照概率 p_m/l 确定是否反转。一次变异有可能改变多个位。

虽然这两种方式实现细节不同,但都具有遍历性,也就是从某个特定的基因型出发,经过有限次迭代可以达到任意一个基因型。

(4) 运行参数

SGA 算法有 4 个运行参数,需要预先设定,即:

- 种群数目 N——种群数目影响 GA 的有效性。N 太小,GA 会很差或根本找不到解,因为太小的种群数目不能提供足够的采样点;N 太大,会增加运算量,使收敛时间过长。一般种群数目在 50~200 之间比较合适。
- 终止条件——一般选择一定的迭代代数或适应度函数值达到一定的阈值作为终止条件。
- 交叉概率 p_c——它控制着交叉操作的频率,决定了个体的更新能力和算法在解空间的搜索能力。p_c 太大,会使优良个体的破坏速度过大,造成算法性能的不稳定;p_c 太小,种群在进化过程中产生具有信息的新个体的速度减慢,搜索会由于太小的探查率而可能停滞不前。一般 p_c 的取值范围为 0.25~0.75。
- 变异概率 p_m——它是增加种群多样性的搜索算子。p_m 太小,不会产生新的基因;p_m 太大,会使 GA 退化成随机搜索。一般 p_m 取值范围为 0.001~0.01。

将 GA 应用于聚类有可能会带来一些问题,如编码冗余、对具体问题不敏感以及随机搜索最优解等。为了克服这些问题,可从编码方案、适应度函数、遗传算子和运行参数等几个方面对 SGA 算法加以改进。作为进化算法中最具有代表性的算法,遗传算法以其简单通用的编码技术和直观有效的进化操作得到了广泛的应用。目前的遗传算法已经不再局限于二进制编码。最近很多的应用尝试使用其他的形式,如图、Lisp 表达式、有序列表和实数向量等。

3）蚁群算法

群体智能(Swarm Intelligence,SI)是一种人工智能技术,主要探讨由多个简单个体构成的种群的集体行为,这些个体之间相互作用,个体与环境之间也是互为影响。尽管没有集中控制机制指导个体的行为,个体之间的局部交互也能够导致某一社会模式的出现。自然界中此类现象很多,如蚁群、鸟群、兽群和蜂群等,由这种自然现象引发的"群类算法",如蚁群算法、粒子群算法能够成功地解决现实中的优化问题。SI 与遗传算法有共同之处,都是基于种群的,系统从一个由多个个体(潜在解)组成的种群开始,这些个体模仿昆虫或动物的社会行为来代代繁殖,以寻求最优。不同于遗传算法的是,群体智能模式不使用交叉和变异这些进化算子,个体只是根据自身与种群中其他个体、与周围环境的关系不断地更新,以求得最优。

蚁群算法是模拟自然界蚂蚁觅食过程的一种分布式、启发式群体智能算法,最早是 1991 年由 Colorni、Dorigo 和 Maniezzo 提出,用于求解复杂的组合优化问题,如旅行商问题(TSP)、加工调度问题(JSSP)和图着色问题(GCP)等。

像蚂蚁这类群居昆虫,虽然单个个体的行为极其简单,但由这样的个体组成的蚁群却表现出极其复杂的行为,能够完成复杂的任务,不仅如此,蚂蚁还能适应环境的变化,如在运动路线上遇到障碍物时,蚂蚁能够很快重新找到最优路径。那么蚁群是如何寻找最优路径的呢?

人们通过大量的研究发现,自然界中的蚂蚁在觅食过程中沿途散播一种化学物质,称为信息素或外激素(Pheromone),信息素中记录了食物源的远近与食物量的多少,而其他蚂蚁通过触角能够检测识别到这种信息素并跟踪,从而最终找到食物源。而当大量蚂蚁不断地从蚁穴通往食物源,沿途不断地识别原有信息素,并同时散播新的信息素,使得越短的路线上的信息素浓度越高,将最终找到一条最短的路线,此后所有的蚂蚁都将走这条最短路径到达食物源。

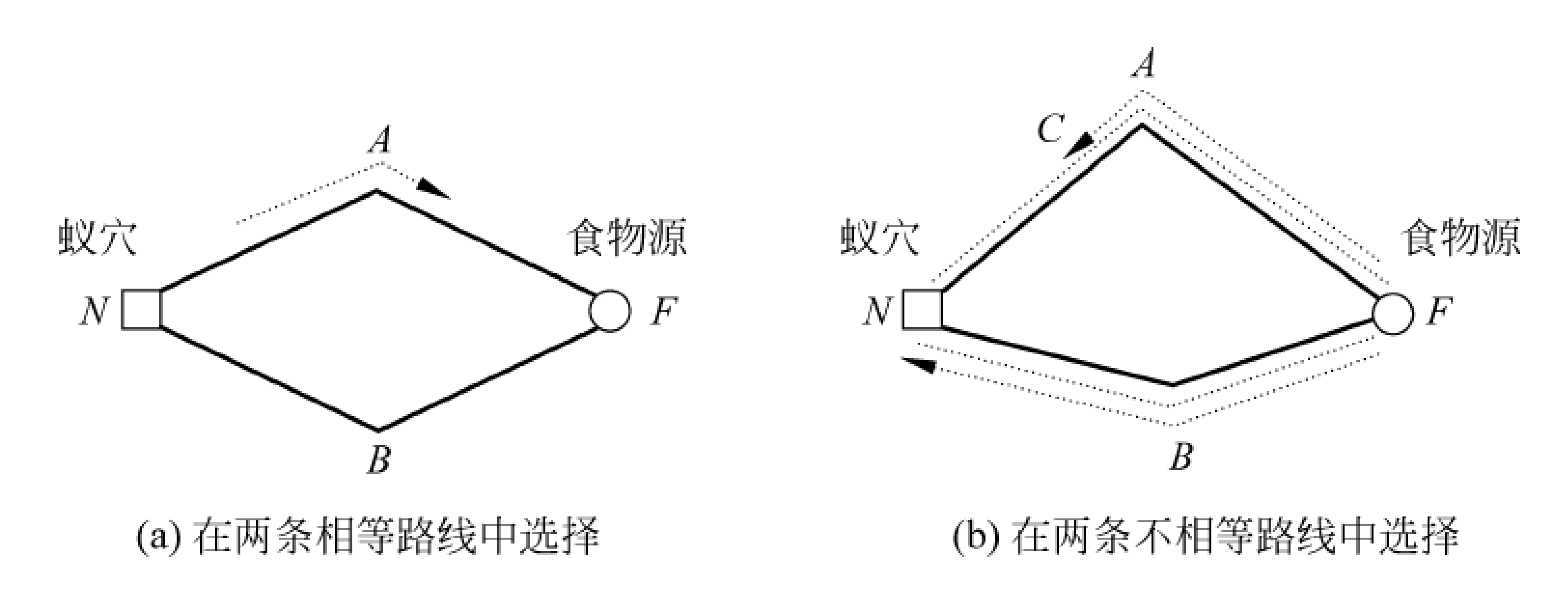


图 6.6　蚂蚁觅食时最短路径选择

如图 6.6(a)所示,假设从蚁穴到食物源有两条等长路线 NAF 和 NBF($NAF=NBF$)。始初时,两条路线上都没有信息素,各个蚂蚁随机选择其中一条路线,并沿途散播信息素;随着时间的推移,各路线会挥发掉部分信息素,也不断地增加新的蚂蚁带来的信息素,这是一个正反馈过程;后来的蚂蚁再选择路线时,浓度较高的路线被选择的概率较大;一段时间后,越来越多的蚂蚁会选择同一条路线,而另一条路线上的蚂蚁数量越来越少,且其上的信息素逐渐挥发殆尽。

如图 6.6(b)所示,对于两条不等长的路线 NAF 和 NBF($NAF>NBF$)而言,初始时两

条路线上都没有信息素，各个蚂蚁随机选择其中一条路线，即有的选择路线 NAF，另一些选择路线 NBF，并沿途散播信息素，两条路线上的蚂蚁数大致相等。假设蚂蚁的行走速度相同，则选择走路线 NBF（较短路线）的蚂蚁比选择走路线 NAF（较长路线）的蚂蚁先到达食物源 F；当走路线 NBF 的蚂蚁返回蚁穴时，走路线 NAF 的蚂蚁仍在途中 C 点处，即 $2NBF=NAF+FAC$。可以看出，线段 NC 上的信息素要少于别处；下次蚂蚁再选择路线时，会以较高概率选择较短路径，这使得较长路线上的信息素浓度越来越低，较短路线上的信息素浓度越来越高。一段时间后，所有的蚂蚁都将选择较短的路线。

蚁群算法就是从蚂蚁觅食时寻找最短路径的现象中得到启示而设计的，由计算机编程实现的分布式并行搜索策略。蚂蚁通过别的蚂蚁留下来的信息素的强弱作为自己选择路径的参数，信息素越强的路径被选择的可能性越大。信息素的更新策略是越好的路径上获得的信息素越多，通过这个正反馈寻找更好的路径，这是蚁群算法的基本原理。单个蚂蚁的规则相当简单，但是通过蚁群的协同工作，产生对复杂环境的认知，实现对解空间的有效搜索。

蚂蚁觅食的过程与旅行商问题非常相似，下面通过求解 n 个城市的 TSP 问题为例说明基本的蚁群算法。

首先设 TSP 中城市 i 与城市 j 之间的距离为 d_{ij}，m 为蚁群中蚂蚁的数量，$b_i(t)$ 表示 t 时刻位于城市 i 的蚂蚁数量，则有 $m=\sum_{i=1}^{n} b_i(t)$。$\tau_{ij}(t)$ 表示 t 时刻弧 (i,j) 上的信息素量。初始时刻各弧上的信息素量相等，$\tau_{ij}(0)=C$，C 为常数。蚂蚁 k 在运动过程中，根据各弧上的信息素量决定移动的方向，$p_{ij}^k(t)$ 表示在 t 时刻蚂蚁 k 由点 i 向 j 移动的概率。

$$p_{ij}^k(t)=\begin{cases}\dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum\limits_{s\in J_k(i)}\tau_{is}^{\alpha}(t)\eta_{is}^{\beta}(t)}, & \text{若 } j\in J_k(i)\\ 0, & \text{否则}\end{cases} \tag{6.30}$$

其中 $J_k(i)$ 表示城市 i 上的蚂蚁 k 下一步允许选择的城市集合。α 和 β 分别表示蚂蚁在移动过程中所积累的信息素 $\tau_{ij}(t)$ 及启发式因子 $\eta_{ij}(t)$ 在蚂蚁择路时的重要程度。η_{ij} 表示由城市 i 到城市 j 的期望值，可模拟某种启发式算法具体确定。另外，蚁群算法还具有记忆功能，用 tab $u_k(k=1,2,\cdots,m)$ 记录蚂蚁 k 当前所走过的城市，集合 tab u_k 随进化过程进行动态调整。随着时间的推移，以前留下的信息素逐渐挥发，用参数 $1-\rho$ 表示信息素挥发程度，经过 l 个时刻，蚂蚁完成一次循环，各弧上的信息素量的调整如下：

$$\tau_{ij}(t+l)=\rho\cdot\tau_{ij}(t)+\Delta\tau_{ij} \tag{6.31}$$

$$\Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^k \tag{6.32}$$

$\Delta\tau_{ij}^k$ 表示第 k 只蚂蚁在本次循环中留在弧 (i,j) 上的信息素量，$\Delta\tau_{ij}$ 表示本次循环中弧 (i,j) 上的信息素的总增量。

$$\Delta\tau_{ij}^k=\begin{cases}\dfrac{Q}{L_k}, & \text{若第 } k \text{ 只蚂蚁在本次循环中经过弧}(i,j)\\ 0, & \text{否则}\end{cases} \tag{6.33}$$

其中，Q 是常数，L_k 表示第 k 只蚂蚁在本次循环中所走路径的总长度。

此模型中，参数 Q,C,α,β,ρ 通常由实验确定其最佳值。

基本蚁群算法求解 TSP 问题的主要步骤如下：

(1) 迭代次数 $nc=0$；各 τ_{ij} 和 $\Delta\tau_{ij}$ 的初始化；将 m 个蚂蚁置于 n 个顶点上。

(2) 将各蚂蚁的初始出发点置于当前解路线集中；对每个蚂蚁 $k(k=1,2,\cdots,m)$，按式(6.30)的概率 p_{ij}^{k} 移至下一顶点 j；将顶点 j 置于当前解路线集中。

(3) 计算各蚂蚁的目标函数值 Z_k；记录当前最好解。

(4) 按更新方程式(6.31)和(6.32)修改轨迹强度。

(5) 对各弧(i,j)，置 $\Delta\tau_{ij}=0$，$nc=nc+1$。

(6) 若 $nc<$预定迭代次数且无退化行为(即找到的都是相同解)，则转步骤(2)。

算法的时间复杂度为 $O(nc\cdot m\cdot n^2)$。就 TSP 问题而言，经验结果是，当 m 约等于 n 时，效果最佳，此时的时间复杂度为 $O(nc\cdot n^3)$。

基本蚁群算法的优点体现在以下几方面。

- 较强的鲁棒性：对基本蚁群算法稍加修改，便可以应用于其他问题。
- 分布式计算：蚁群算法是一种基于种群的进化策略，具有并行性，易于并行实现。
- 易于与其他方法结合：蚁群算法很容易与多种启发式算法结合，以改善算法的性能。

研究证明蚁群算法具有很强的发现较好解的能力，这是因为该算法不仅利用了正反馈原理，在一定程度上可以加快进化过程，而且是一种并行算法，不同个体之间不断进行信息交流和传递，能够相互协作，有利于发现较好解。

蚁群算法也存在一些缺陷，例如：

- 搜索时间较长：与其他算法相比，一般需要较长的搜索时间，其算法的复杂度可以反映这一点。
- 易出现停滞现象：当搜索到一定程度后，所有个体所发现的解完全一致，不能对解空间进一步搜索，不利于发现更好的解。

目前，蚁群算法是继遗传算法、模拟退火、禁忌搜索和神经网络等热门算法之后，新加入智能启发式算法这一行列的，在短时间内，受到越来越多的关注，作为通用型随机优化方法，通过其内在的搜索机制，在一系列困难的组合优化问题求解中取得了成效。

4) 粒子群优化算法

粒子群优化算法(Patical Swarm Optimization，PSO)最初由 Kennedy 和 Eberhart 于 1995 年提出，是一种基于迭代的优化方法，因其概念简单、实现容易迅速引起重视。目前已被应用于多目标优化、模式识别、信号处理和决策支持等领域。

PSO 最早源于对鸟群觅食行为的研究，与蚁群算法同属于群体智能算法，是从个体的社会行为中得到启示，是对简单社会系统的模拟。PSO 算法中，粒子群在一个 n 维的空间中搜索，其中的每一个粒子所处的位置都表示问题的一个解。粒子通过不断调整其位置 X 搜索新的解。每个粒子都能记住自己搜索到的最好解，记作 P_{id}，以及整个粒子群经历过的最好位置，即目前搜索到的最优解，记作 P_{gd}。每个粒子都具有一个速度，记做 V，由公式(6.34)计算：

$$V'_{id}=\omega\cdot V_{id}+\eta_1\cdot\text{rand}()\cdot(P_{id}-X_{id})+\eta_2\cdot\text{rand}()\cdot(P_{gd}-X_{id}) \qquad (6.34)$$

其中 V_{id} 表示第 i 个粒子第 d 维上的速度，ω 为惯性权重，η_1 和 η_2 为调节 P_{id} 和 P_{gd} 相对重要性的参数，rand()为随机数生成函数。这样，可以计算出粒子移动的下一位置为：

$$X'_{id} = X_{id} + V_{id} \tag{6.35}$$

从公式(6.34)和(6.35)可以看出,粒子的移动方向由三部分决定,自己原有的速度 V_{id}、与自身经历的最佳距离($P_{id}-X_{id}$)和与群体经历的最佳距离($P_{gd}-X_{id}$),并分别由权重系数 ω、η_1 和 η_2 决定其相对重要性。

标准 PSO 算法的步骤如下:

(1) 初始化粒子群,即随机设定各粒子的初始位置 X 和初始速度 V。

(2) 计算每个粒子的适应度值。

(3) 对每个粒子,将它的适应度值和它经历过的最好位置 P_{id} 的适应度值进行比较,如果更好,则更新 P_{id}。

(4) 对每个粒子,将它的适应度值和种群所经历最好位置 P_{gd} 的适应度值进行比较,如果更好,则更新 P_{gd}。

(5) 根据式(6.34)和式(6.35)调整粒子的速度和位置。

(6) 如果达到结束条件(足够好的位置或最大迭代次数),则结束;否则转步骤(2)。

PSO 是一种进化计算方法,具有以下进化计算的典型特征。

- 具有一个初始化过程,在这一过程中,群体中的个体被赋值为一些随机产生的初始解;
- 通过产生更好的新一代群体搜索解空间;
- 新一代群体产生在前一代的基础之上。

目前,PSO 算法在很多连续优化问题中得到较成功的应用,但是在离散域上的研究和应用还很少。

6.2 模糊聚类

6.2.1 概述

通常地,硬聚类是指将包含 n 个对象的集合划分为 k 个互斥的类,聚类结果可表示为一个 $n\times k$ 的矩阵 $\boldsymbol{U}=(u_{ik})$,若对象 i 属于类 k,则 $u_{ik}=1$;否则 $u_{ik}=0$。为保证各类是分离且非空的,u_{ik} 必须满足下列条件:

$$\sum_{k=1}^{K} u_{ik} = 1 \qquad i = 1,\cdots,n$$

$$\sum_{i=1}^{n} u_{ik} > 0 \quad k = 1,\cdots,K \tag{6.36}$$

其中,$u_{ik}\in\{0,1\}$　$i=1,\cdots,n$;$k=1,\cdots,K$。

实际上,互斥的聚类在实际中并不适合,因此提出了模糊聚类,与硬聚类的本质区别在于:

$$u_{ik} \in [0,1] \qquad i = 1,\cdots,n;\ k = 1,\cdots,K \tag{6.37}$$

模糊聚类(Fuzzy Clustering Analysis,FCA)是指一个对象以不同程度属于多个类,各个类之间的界限是不确定的。其本质是不仅要考虑对象是否属于该类,而且要考虑属于该类的程度如何。模糊聚类完全不同于所谓的硬聚类,即类别之间的界限是明确而严格的。

1966年,Bellman、Kalaba和Zadeh首先提出以模糊集为基础实现聚类。其后,Wee、Flake和Turner及Gitman和Levine等人进行了一些尝试性探索,系统地阐述模糊聚类算法的是著名学者Ruspini。20世纪70年代到80年代,人们对模糊矩阵及其传递闭包等问题进行了大量研究;到20世纪90年代,尽管仍有人从事这一方面的研究,但由于这类方法不适用于大型数据集合,所以日渐冷落。后来人们试图用图论的方法研究模糊聚类,1993年Zhenggu Wu和Leathy提出最优图论的聚类方法。在将硬聚类推广到模糊聚类方面进行了大量的工作,如将k-最近邻推广到模糊聚类,1986年Bezdek等人将模糊c-均值应用到k-最近邻,提出了一种模糊k-最近邻法。此外还提出了其他一些模糊聚类方法,如1987年Bezdek和Harri利用数据集的凸分解进行模糊聚类。

由于种种原因,上述方法的应用并不广泛。实际中,受到普遍关注的是基于目标函数的模糊聚类如模糊c-均值等。

基于目标函数的模糊聚类算法首先是由Ruspini提出的,随后其一般化方法——模糊c-均值及其收敛性被提出并加以证明。从此,基于目标函数的模糊聚类方法得以迅速发展,目前已形成了庞大体系。它是近年来发展很快的一种聚类方法。其目的是使用模糊系统解决客观世界中存在的界限不分明的聚类问题,对样本进行合理的模糊划分,从而达到判别、分析和预测的目的。

模糊聚类的研究工作大致可分为两类:

(1) 通过模糊方法得到模糊结果,每个模式以不同的隶属度从属于若干个类。以Bezdek提出的模糊k-均值为代表,该算法收敛于部分最优,存在局部极值问题。Al. Sultan等人采用模拟退火算法使结果收敛于全局最优,它们均为迭代算法。

(2) 通过模糊方法得到确定的结果,每个模式仅从属于特定的类,如Miyamoto等人提出的算法,需要反复计算分类对象之间的模糊相似度。

目前,有关模糊聚类的研究大多是对模糊c-均值的推广与改进。

大致地,模糊聚类算法包括系统聚类法、传递闭包法、最小支撑树(如Prim和Kruskal算法)、动态直接聚类算法、模糊c-均值和人工神经网络等。

模糊聚类中,数据对象对各个类的隶属度值介于[0,1]区间。模糊聚类考虑到了对象之间的联系,认为每一对象与各聚类中心都存在着一定的隶属关系。模糊聚类能够有效地对类与类之间存在交叉的数据集进行聚类,所得的聚类结果明显优于传统聚类。一般地,模糊聚类要求每一对象对各类的隶属度之和为1,这一约束是对划分情况的概率约束。但是,这一约束无法反映对象的典型性,对含有噪声的数据集合的聚类很不理想,与传统聚类相比,通常模糊聚类算法的收敛速度要慢。

模糊聚类方法已被广泛应用到数据挖掘、模式识别、机器学习以及决策支持等领域。例如可依据“体重/身高”,把人分成“胖人集”、“不胖不瘦集”和“瘦人集”等;对超市可根据“月底销售数量/月初库存”,确定出“畅销商品集”与“滞销商品集”,并进一步分析在同一类别内的商品之间的销售相关性,以支持营销决策等。此外,模糊聚类还可应用到对多维图像的识别与分割、天然中草药分类等。

设数据集合为$X=\{x_1,x_2,\cdots,x_n\}$,$\forall x_i\in X$,其对象$x_i=(x_{i1},x_{i2},\cdots,x_{ip})$,$x_{ik}$表示第$i$个对象的第$k$个属性。模糊相似矩阵(Fuzzy Dissimilarity Matrix)$(r_{ij})_{n\times n}$用于存储n个对象彼此之间的模糊相似度,是一个$n\times n$维的对角线元素为1的对称矩阵,即:

$$\begin{bmatrix} 1 & & & & \\ r_{21} & 1 & & & \\ r_{31} & r_{32} & 1 & & \\ \vdots & \vdots & \vdots & & \\ r_{n1} & r_{n2} & r_{n3} & \cdots & 1 \end{bmatrix} \tag{6.38}$$

常用的计算模糊相似度的方法包括：

(1) 数量积法

$$r_{ij} = \begin{cases} 1, i = j \\ \dfrac{1}{M}\sum_{k=1}^{m} x_{ik}x_{jk}, i \neq j \end{cases} \tag{6.39}$$

其中，$M = \max\limits_{i \neq j}\left(\sum_{k=1}^{m} x_{ik}x_{jk}\right)$。如果 r_{ij} 为负值，可用公式(6.40)修正。

令
$$r'_{ij} = \frac{r_{ij}+1}{2} \text{ 则 } r'_{ij} \in [0,1] \tag{6.40}$$

(2) 夹角余弦法

$$r_{ij} = \frac{\sum_{k=1}^{m} x_{ik}x_{jk}}{\sqrt{\sum_{k=1}^{m} x_{ik}^2}\sqrt{\sum_{k=1}^{m} x_{jk}^2}} \tag{6.41}$$

(3) 统计相关系数法

$$r_{ij} = \frac{\sum_{k=1}^{m}(x_{ik}-\bar{x}_i)(x_{jk}-\bar{x}_j)}{\sqrt{\sum_{k=1}^{m}(x_{ik}-\bar{x}_i)^2}\sqrt{\sum_{k=1}^{m}(x_{jk}-\bar{x}_j)^2}} \tag{6.42}$$

其中，$\bar{x}_i = \frac{1}{m}\sum_{k=1}^{m} x_{ik}, \bar{x}_j = \frac{1}{m}\sum_{k=1}^{m} x_{jk}$。

(4) 最大最小法

$$r_{ij} = \frac{\sum_{k=1}^{m}(x_{ik} \wedge x_{jk})}{\sum_{k=1}^{m}(x_{ik} \vee x_{jk})} \tag{6.43}$$

(5) 算术平均法

$$r_{ij} = \frac{2\sum_{k=1}^{m}(x_{ik} \wedge x_{jk})}{\sum_{k=1}^{m}(x_{ik} + x_{jk})} \tag{6.44}$$

(6) 几何平均最小法

$$r_{ij} = \frac{\sum_{k=1}^{m}(x_{ik} \wedge x_{jk})}{\sum_{k=1}^{m}\sqrt{x_{ik}x_{jk}}} \tag{6.45}$$

(7) 绝对值指数法

$$r_{ij} = e^{-\sum_{k=1}^{m}|x_{ik}-x_{jk}|} \tag{6.46}$$

(8) 指数相似系数法

$$r_{ij} = \frac{1}{m}\sum_{k=1}^{m} e^{-(x_{ik}-x_{jk})^2} \tag{6.47}$$

(9) 绝对值倒数法

$$r_{ij} = \begin{cases} 1, & i = j \\ \dfrac{M}{\sum_{k=1}^{m}|x_{ik}-x_{jk}|}, & i \neq j \end{cases} \tag{6.48}$$

适当选取 M,使 $r_{ij} \in [0,1]$。

(10) 绝对值减数法

$$r_{ij} = 1 - c\sum_{k=1}^{m}|x_{ik}-x_{jk}| \tag{6.49}$$

适当选取 c,使 $r_{ij} \in [0,1]$。

6.2.2 主要算法

下面介绍几种主要的模糊聚类算法。

1. 模糊 *c*-均值

目前,模糊聚类算法中应用最广泛而且较成功的是 1974 年由 Dunn 提出并由 Bezdek 加以推广的模糊 c-均值(Fuzzy C-Means,FCM)。

假设待聚类样本数为 n,聚类数为 c,特征数为 s,则有如下定义:

定义 6.1:样本集 $X=\{x_1,x_2,\cdots,x_n\}$是任一有限集,$X\subset R^s$,V_{cn}是 $c\times n$ 阶实矩阵的集合,c 是整数($2\leqslant c<n$),则称下述集合为 X 的模糊 c-均值划分空间:

$$M_{fc} = \{U \in V_{cn} \mid 0 \leqslant u_{ik} \leqslant 1, \forall i,k; \sum_{i=1}^{c} u_{ik} = 1, \forall k; 0 \leqslant \sum_{k=1}^{n} u_{ik} < n, \forall k\} \tag{6.50}$$

其中 u_{ik}是隶属度矩阵 $\boldsymbol{U}\in M_{fc}$的 i 行 k 列元素,表示 x_k 对于类 i 的隶属度值。

定义 6.2:设 $v_i\in R^s$ 是类别 i 的聚类典范值(中心)向量,定义 c 聚类典范值矩阵为:

$$\boldsymbol{V} = (v_1,v_2,\cdots,v_c)^{\mathrm{T}} \tag{6.51}$$

模糊 c-均值的目标函数 J_m 定义为:

$$J_m(\boldsymbol{U},\boldsymbol{V}) = \sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})^m(d_{ik})^2 \tag{6.52}$$

其中$(d_{ik})^2 = \| x_k - v_i \|^2$,$\| \cdot \|$ 是 R^s 上任一内积导出的范数;m 称加权指数,$m\in[1,\infty)$。

目标函数的值是样本数据到 c 聚类典范值的平方距离的加权累计和,权重由 m 和模糊隶属度函数值共同确定。J_m 反映在某种差异性定义下的类内紧致度。J_m 越小,聚类越紧致;而 m 越大,紧致度的模糊性越大,因为 m 控制隶属度在各类之间共享的程度。

FCM 算法步骤如下:

(1) 设定迭代停止阈值 ε_L 为一个小的正数;初始化迭代次数为 $l=0$ 和 $U^{(0)}$;

(2) 将 $\boldsymbol{U}^{(1)}$ 代入公式(6.51)计算 c-聚类典范值矩阵 $\boldsymbol{V}^{(1)}$：

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m}, \forall i \tag{6.53}$$

(3) 利用 $\boldsymbol{V}^{(1)}$ 更新 $\boldsymbol{U}^{(1)}$，其过程如下：

$\forall$ 样本 x_k，计算类别标号的集合：$I_k=\{i \mid 1\leqslant i\leqslant c; d_{ik}=0\}$ 和 $\tilde{I}_k=\{1,2,\cdots,c\}-I_k$；

如果 $$I_k = \varnothing, 则\ u_{ik} = \frac{1}{\sum_{j=1}^{c} (d_{ik}/d_{jk})^{2/(m-1)}} \tag{6.54}$$

否则 $$u_{ik} = 0, \forall i \in \tilde{I}_k, 且 \sum_{i\in \tilde{I}_k} u_{ik} = 1 \tag{6.55}$$

选用合适的矩阵范数比较 $\boldsymbol{U}^{(l)}$ 和 $\boldsymbol{U}^{(l+1)}$，如果 $\{\boldsymbol{U}^{(l+1)}-\boldsymbol{U}^{(l)}\}\leqslant \varepsilon_L$，则停止迭代；否则 $l=l+1$，返回步骤(2)。

在 FCM 算法中，模糊指数 m 的取值是关键问题。m 值越大，聚类的范围越大，隶属函数的模糊程度越大。当 m 趋近 1 时，FCM 相当于硬 c-均值聚类算法，所以 m 不应该趋近 1。目前还没有发现确定合适 m 的方法。对于 FCM，根据经验一般取 $1.1\leqslant M\leqslant 5$。Pal 等人从聚类有效性方面研究得到 m 的最佳取值区间为[1.5,2.5]，且一般 $m=2$。

FCM 的本质是使聚类中所有样本到聚类中心的距离平方和最小。通过优化模糊目标函数得到每一样本对类中心的隶属度。但是，FCM 有时会收敛到局部极小值，这限制了 FCM 算法的应用。

2. 模糊关系传递闭包

设有 n 个样本 $U=(x_1,x_2,\cdots,x_n)$，其中每个样本具有 m 个特征，即 $x_i=(x_{i1},x_{i2},\cdots,x_{im})$。利用多元分析方法建立样本之间的模糊关系 $R:U\times U\to[0,1]$，$(x_i,x_j)\in R$ 的程度通常用隶属函数 $\mu_R(x_i,x_j)$ 表示，$\mu_R(x_i,x_j)$ 被称为 x_i 和 x_j 的相似度，满足 $0\leqslant\mu_R(x_i,x_j)\leqslant 1$，$\mu_R(x_i,x_j)=\mu_R(x_j,x_i)$。$\mu_R(x_i,x_j)$ 越大，两者的相似度越高，定义 $\mu_R(x_i,x_i)=1, i=1,2,\cdots,n$。模糊关系的确定采用距离法，例如切比雪夫距离 $\mu_R(x_i,x_j)=1-c\max\limits_{1\leqslant i,j\leqslant m}|x_{ik}-x_{jk}|$，其中 c 表示一个常数，或者海明距离、欧氏距离和相似系数等。由 n 个样本之间的模糊关系形成一个 $n\times n$ 矩阵 $\boldsymbol{R}=(\boldsymbol{\mu}_R(x_i,x_j))_{n\times n}$，称 $\boldsymbol{R}$ 为模糊相似矩阵。

设 U、V 和 W 分别表示三个论域，若 R、S 分别为 $U\times V$ 和 $V\times W$ 上的模糊关系，定义关系的合成 $R\circ S$ 为 $U\times W$ 上的模糊关系，其隶属度函数定义为：

$$\mu_{R\circ S}(x,z) = \{\max_{y\leqslant V}\{\min[\mu_R(x,y),\mu_S(y,z)]\}\} \tag{6.56}$$

其中，$\forall x\in U$，$\forall z\in W$。

若 R 是 U 上的模糊关系，并满足 $R\circ R\subseteq R$，则称 R 为模糊传递矩阵。

若模糊相似矩阵 R 满足传递性，则称 R 为等价关系。对任意的 $\alpha\in[0,1]$，集合 $R_\alpha=\{(x,y)\mid\mu_R(x,y)\geqslant\alpha\}$ 称为 R 的 α-截集，α 称为截集阈值。若 $\boldsymbol{R}$ 为模糊等价关系，则对任意 $\alpha\in[0,1]$，$\boldsymbol{R}_\alpha$ 也为模糊等价关系，有时称 $\boldsymbol{R}_\alpha$ 为 α-截矩阵。若 $\boldsymbol{R}_\alpha$ 为等价矩阵，则 $\forall x\in U$，$[x]=[y\mid\mu_R(x,y)\geqslant\alpha]$ 构成 U 相对于阈值 α 的模糊聚类。

模糊相似矩阵 $\boldsymbol{R}$ 的传递闭包是指包含 $\boldsymbol{R}$ 的最小模糊等价矩阵。利用平方法可以求得模糊相似矩阵的传递闭包。

定理 6.1：设 $\boldsymbol{R}$ 为 n 阶模糊相似矩阵，则存在一个最小自然数 $k(k<n)$，使得传递闭包 $t(\boldsymbol{R})=\boldsymbol{R}^k$，且对一切大于 k 的自然数 L 恒有 $\boldsymbol{R}^L=\boldsymbol{R}^k$。

定理 6.1 说明从模糊相似矩阵 $\boldsymbol{R}$ 开始，依次计算 R^2、R^4、$\boldsymbol{R}^8\cdots$直到出现 $\boldsymbol{R}^k\cdot\boldsymbol{R}^k=\boldsymbol{R}^k$ 时，$\boldsymbol{R}^k$ 就是传递包 $t(\boldsymbol{R})$。设 $\boldsymbol{R}$ 和 $\boldsymbol{S}$ 为模糊相似矩阵，$\boldsymbol{R}=(r_{ij})_{n\times n}$，$\boldsymbol{S}=(s_{ij})_{n\times n}$，则 $\boldsymbol{R}\cdot\boldsymbol{S}=(t_{ij})_{n\times n}$，其中，$t_{ij}=\bigvee_{k=1}^{n}(r_{ik}\wedge s_{kj})$。

基于模糊关系传递闭包的模糊聚类算法的步骤如下：

(1) 确定集合 $X=\{x_1,x_2,\cdots,x_n\}$上模糊相似矩阵 $\boldsymbol{R}$ 和一个截集阈值 α；

(2) 采用自乘法将 $\boldsymbol{R}$ 按如下计算构造为一个模糊等价矩阵：

$$\boldsymbol{R}\cdot\boldsymbol{R}=\boldsymbol{R}^2$$
$$\boldsymbol{R}^2\cdot\boldsymbol{R}^2=\boldsymbol{R}^4$$
$$\cdots$$

直到存在 k，满足 $\boldsymbol{R}^{2k}=\boldsymbol{R}^k$，则 $\boldsymbol{R}^k$ 即为一个模糊等价矩阵。

(3) 计算模糊等价矩阵的 α—截矩阵 $\boldsymbol{R}_\alpha=(r_{ij\alpha})_{n\times n}$，且

$$r_{ij\alpha}=\begin{cases}1 & r_{ij}\geqslant\alpha\\0 & r_{ij}<\alpha\end{cases}$$

(4) 输出模糊聚类结果，即将 α 截矩阵中相同行的样本归为一类，表示为 $[x]=\{y\mid\boldsymbol{R}^*(x,y)\geqslant\alpha\}$；

(5) 若满足聚类终止条件，则停止；否则，改变截集阈值 α，转至步骤(2)。

该算法中将模糊相似矩阵改造成模糊等价矩阵采用自乘法，计算时间较长，算法计算复杂度为 $O(n^3\log_2 n)$。

3. 最小支撑树

最小支撑树算法的步骤如下：

(1) 建立模糊相似矩阵；

(2) 构建最小支撑树；

(3) 聚类。

典型的最小支撑树算法包括 Prim 法和 Kruskal 法。

设待聚类对象集合为{1,2,3,4,5}，给定如下的模糊相似矩阵：

$$\boldsymbol{R}=\begin{bmatrix}1 &&&&\\0.1 & 1 &&&\\0.8 & 0.1 & 1 &&\\0.5 & 0.2 & 0.3 & 1 &\\0.3 & 0.4 & 0.1 & 0.6 & 1\end{bmatrix}$$

1) Prim 算法

(1) 先取对象 1，在对象 2、3、4 和 5 中，找出与对象 1 相似度最大的，可得

$0.8=\boldsymbol{R}(1,3)$，得到：

$1\xrightarrow{0.8}3$

在对象 2、4 和 5 中，找到与对象 1 相似度最大的 $0.5=\boldsymbol{R}(1,4)$，找出与对象 3 相似度最大的 $0.3=R(3,4)$，因 $0.5>0.3$，取对象 4，得到：

$4\xrightarrow{0.5}1\xrightarrow{0.8}3$

再在对象 2 和 5 中，找出与对象 1、3 和 4 相似度最大的 $0.6=\boldsymbol{R}(4,5)$，得到：

$5\xrightarrow{0.6}4\xrightarrow{0.5}1\xrightarrow{0.8}3$

最后，找出对象 2 与 1、3、4 和 5 之间相似度最大的 $0.4=\boldsymbol{R}(2,5)$，得最小支撑树，即

$2\xrightarrow{0.4}5\xrightarrow{0.6}4\xrightarrow{0.5}1\xrightarrow{0.8}3$

(2) 取 $\lambda\in[0,1]$，砍断连接权重小于 λ 的枝，可以得到一个不连通的图，而各连通分支就构成了 λ 水平上的分类。

若取 $\lambda\in[0,0.4]$，则只得一类{1,2,3,4,5}；若取 $\lambda\in(0.4,0.5)$，则得两类{2}、{1,3,4,5}；若取 $\lambda\in(0.5,0.6)$，则得到三类{2}、{4,5}和{1,3}；若取 $\lambda\in(0.6,0.8)$，则得四类{2}、{5}、{4}和{1,3}；若取 $\lambda\in(0.8,1)$，则得五类{1}、{2}、{3}、{4}和{5}。

2) Kruskal 算法

(1) 先在 $\boldsymbol{R}$ 的非主对角线中找到最大元 $0.8=\boldsymbol{R}(1,3)$，得到：

$3\xrightarrow{0.8}1$

再找次最大元 $0.6=\boldsymbol{R}(4,5)$，得到：

$3\xrightarrow{0.8}1\qquad 4\xrightarrow{0.6}5$

而后找到 $0.5=\boldsymbol{R}(1,4)$，得到：

$3\xrightarrow{0.8}1\xrightarrow{0.5}4\xrightarrow{0.6}5$

最后得到 $0.4=\boldsymbol{R}(2,5)$，至此所有顶点都被连到，且不含圈，得到最小支撑树，即

$3\xrightarrow{0.8}1\xrightarrow{0.5}4\xrightarrow{0.6}5\xrightarrow{0.4}2$

(2) 同 Prim 算法中的步骤(2)。

采用上述两种方法所得的最小支撑树可能不同，但可以证明其聚类结果相同。其中，Prim 算法的计算复杂度最多为 $O\left(\frac{3n^3}{2}\right)$，Kruskal 算法的计算复杂度最多为 $O(n^3\log_2 n)$。

6.3 评价

聚类分析是一种无监督的学习，事先对给定数据集合的结构一无所知，没有利用任何先验知识。无论采用哪种聚类算法，其聚类结果的合理性和有效性都有待评价。聚类有效性评价对聚类分析具有重要意义。对于相同的数据集合，采用不同的聚类方法，可能得到不同的聚类结果。即便是采用同一种聚类方法，若初始参数（如聚类数、聚类中心等）选择不同也可能会得到不同的结果。例如，采用同一种 k-均值聚类算法对同一个 Wine 测试数据集（来自 UCI 机器学习数据库）进行聚类，当预设聚类类别数分别为 1～8 时，则所得到的聚类正确率是不同的，如图 6.7 所示。

此外，对于基于目标函数的聚类算法，迭代优化易收敛于局部极值，难以获得全局最优解。以 k-均值为例，常采用梯度下降法实现迭代。由于梯度法的搜索方向是沿着能量减小

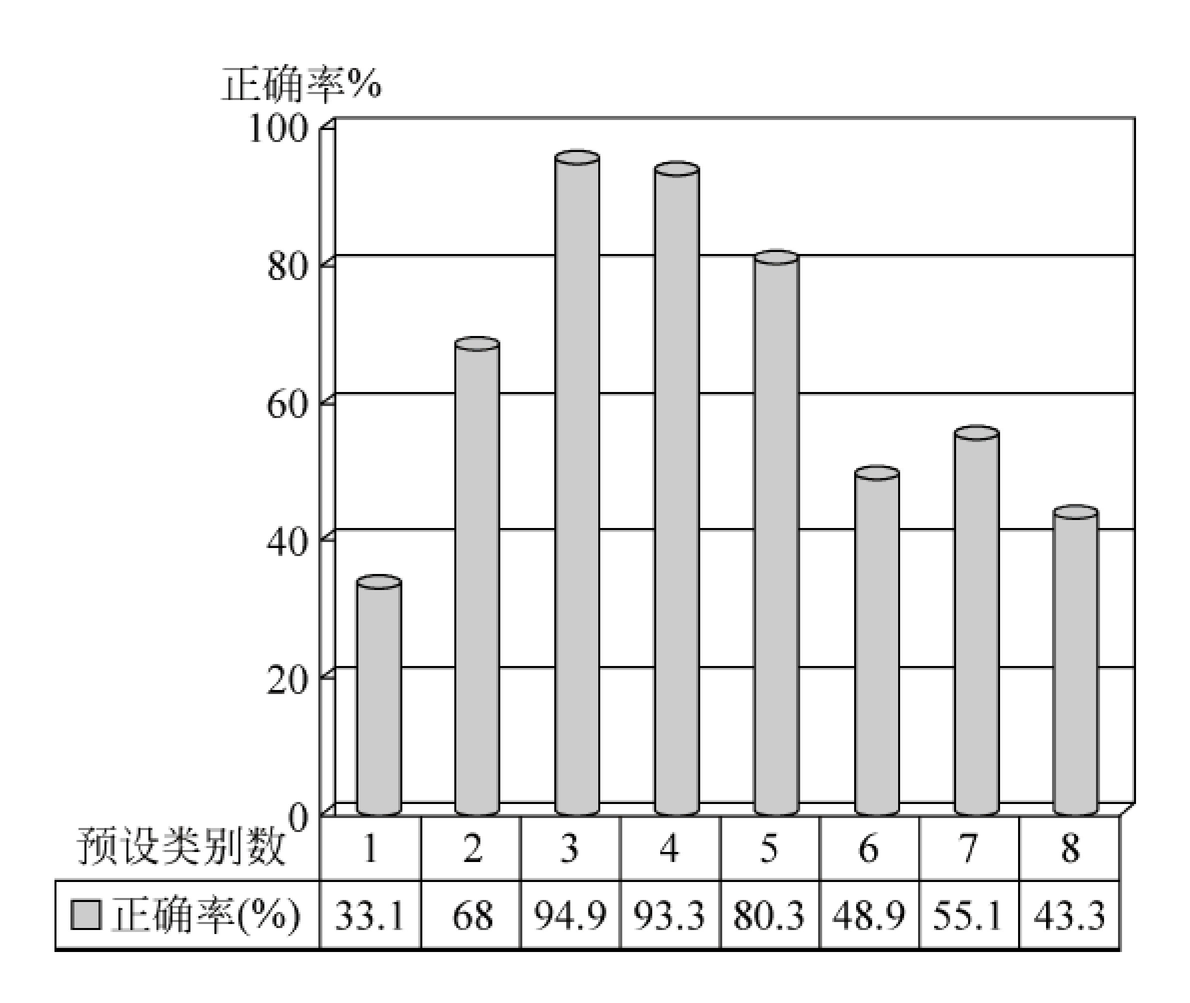


图 6.7　不同的聚类数初值导致 k-均值的聚类正确率不同

的方向进行,因此易陷入局部极值。例如,对于如图 6.8 所示的由 12 个对象组成的集合,显然不管采用什么聚类方法,其结果均应为 3 类,结论应该是一致的。然而,理论分析和仿真实验均表明,使用 k-均值求解这一简单问题时,经常会出现错解或无解的情况。若按横坐标从小到大将对象依次编号为 1～12,当 3 个初始聚类中心分别选取 2、3 和 6,或 6、10 和 11,或 1、4 和 8 时,算法均得出错误的结论;当 3 个初始聚类中心分别选取 1、4 和 10 时,算法得不到任何解。为了避免出现上述问题,近年来提出了多种算法以提高目标函数的优化效率。例如,1999 年 Krishma 根据遗传算法的原理以 k-均值算子代替遗传算法中的交叉算子,提出了一种混合遗传算法;2000 年 Mali 采用聚类中心的浮点编码方法,并设计了浮点数交叉和变异算子,从而提高聚类算法的搜索效率。但是,仿真实验结果表明,当聚类数目、对象个数和维度较大时,这两种基于遗传算法的聚类方法经常会出现早熟现象。聚类规模越大,早熟越容易发生,而且由于进化算法在进化过程中可能产生退化现象,从而导致迭代次数过多以及聚类准确率不高,并且可能出现进化后期的波动现象;2003 年行小帅等提出了基于免疫规划的 k-均值聚类算法,其主要思想是在合理提取免疫疫苗的基础上,通过接种疫苗和选择操作分别提高个体的适应度并防止种群的退化,从而有效地避免局部极值,提高全局寻优性。

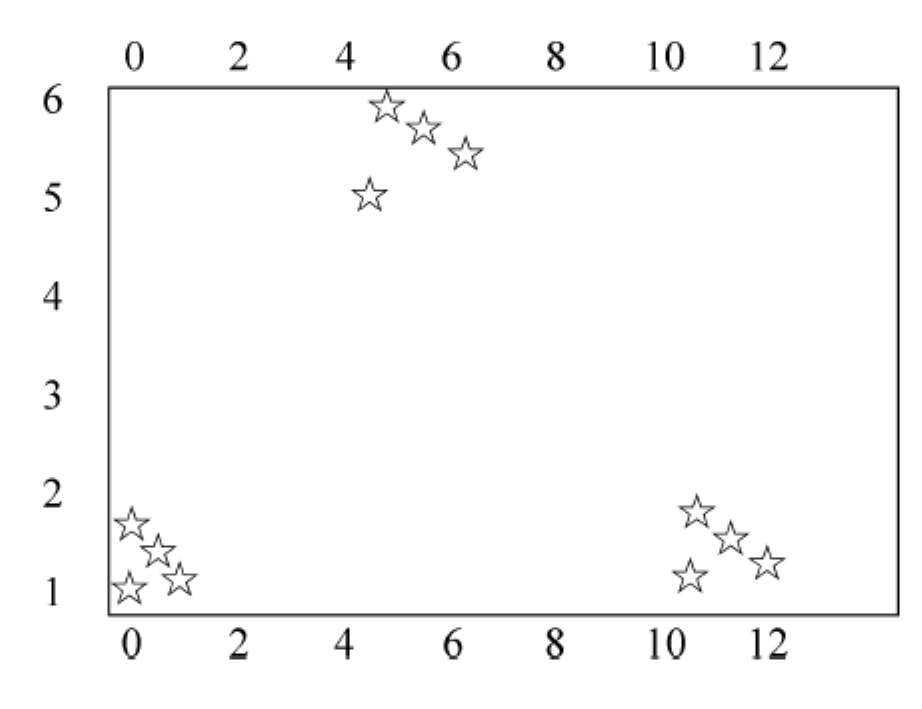


图 6.8　k-均值算法失效的示例

如何评价不同聚类算法的性能呢？评价聚类算法的一般标准包括：

(1) 可伸缩性

即算法中模式数增大的情况。有些算法在数据量小的情况下，算法的性能很好，但是数据量增大后，算法性能下降。如 k-中心点，它对小的数据集合非常有效，但对大的数据集合不具有良好的可伸缩性。

(2) 高维性

有些算法只擅长处理低维数据。高维空间的聚类是一个挑战，特别是非常稀疏和偏斜的数据。

(3) 发现任意形状的聚类

一个簇可能是任意形状的，但一般的聚类算法是基于欧氏距离和曼哈顿距离实现聚类，更趋于发现球状簇，在这方面基于密度的方法较好。

(4) 处理噪声的能力

噪声可能是数据本身不完整，也可能是孤立点。有些算法不擅于处理孤立点数据，由此还专门出现了发现孤立点的算法。

(5) 输入参数的领域知识最小化和输入顺序敏感性

一方面要求降低算法对输入参数的敏感度，另一方面要求输入顺序对算法结果的影响小。如 k-均值算法，需要预先给出簇的数目，这一参数非常影响聚类效果，这常常是高效率算法的弱点。

(6) 可解释性和可用性

知识发现过程中，聚类结果需要表现为知识，即要求聚类结果可解释、易理解。这与可视化密切相关，同时也与实际应用有关。如 SOM 算法用于文本聚类可以产生知识地图，具有良好的可视化功能。

1965 年，Zadeh 首次给出聚类有效性的度量——分离度(Degree of Separation)。但是，后来发现它对模糊聚类有效性的判断并不十分有用；1974 年 Bezdek 提出划分系数(Partition Coefficient)，这是第一个有用的度量聚类有效性的泛函，旨在度量各聚类之间的“重叠”程度。

定义 6.3：设 $U \in M_{fc}$ 为集合 X 的模糊 c-划分，$|X|=n, 2 \leqslant c < n$，则 U 的划分系数是：

$$F(U;c) = \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^2 / n \tag{6.57}$$

其中，n 为数据集的样本个数，c 为模糊聚类个数，u_{ik} 为数据点 k 属于类 i 的模糊隶属度。若所有的 u_{ik} 接近 0 或 1，则熵小，所给出的聚类效果较好；若 u_{ik} 接近 0.5，则聚类的模糊程度高，则熵大，聚类效果较差。最佳有效划分是 $\max_c\{\max_{\Omega_c}\{F(U;c)\}\}$。其主要缺点是具有单调下降趋势以及与数据集本身的特征缺乏直接的关联。

类似地，Shannon 提出了划分熵(Classification Entropy)。

定义 6.4：设 $U \in M_{fc}$ 为集合 X 的模糊 c-划分，$|X|=n, 1 \leqslant c < n$，则 U 的划分熵是：

$$H(U;c) = -\sum_{j=1}^{n} \sum_{i=1}^{c} (\mu_{ij})^2 \log_a (u_{ij}) / n \tag{6.58}$$

其中，$a \in (1, \infty)$ 且只要 $u_{ij}=0$ 则 $u_{ij}\log_a(u_{ij})=0$。最佳有效划分是 $\min_c\{\min_{\Omega_c}\{H(U;c)\}\}$。

划分系数与划分熵两者之间的关系可表示为：

$$0 \leqslant 1 - F(U;c) \leqslant H(U;c)/\log_a(e) \tag{6.59}$$

其中，$a \in (1, \infty)$ 且 e=2.718…。

为了克服上述两种有效性函数的不足，Windham 提出比例指数（Proportion Exponent）。

定义 6.5：设 $U \in (M_{fc} - M_{\infty})$ 为集合 X 的模糊 c-划分，$|X| = n, 2 \leqslant c < n$，对于 U 的 k 列，$1 \leqslant k \leqslant n$，使

$$u_k = \max_{1 \leqslant i \leqslant c}\{u_{ik}\} = \bigvee_{i=1}^{c} u_{ik} \tag{6.60}$$

$$[u_k^{-1}] = \text{greatestinteger} \leqslant (1/u_k) \tag{6.61}$$

则 U 的比例指数是：

$$P(U;c) = -\log_e\left\{\prod_{k=1}^{n}\left[\sum_{j=1}^{k}(-1)^{j+1}\binom{c}{j}^{[\mu-1]}\left(1 - j \cdot \bigvee_{i=1}^{c} u_{ik}\right)^{c-1}\right]\right\} \tag{6.62}$$

最佳有效划分是 $\max\limits_{2 \leqslant c \leqslant n}\{\max\limits_{\Omega_c}\{P(U;c)\}\}$。

后来，Gunderson 提出分离系数(Separation Coefficient)，考虑了数据集的几何特征，但无法直接应用于模糊聚类，必须首先将模糊聚类转换为硬聚类。

直到 1989 年 Xie 和 Beni 提出了 Xie-Beni 方法，才使聚类有效性问题得以进一步的发展。

设数据集合 $X = \{X_j, j = 1, 2, \cdots, n\}$ 具有 c 个模糊划分，$V_i (i = 1, 2, \cdots, c)$ 为各聚类中心，$\mu_{ij} (i = 1, 2, \cdots, c; j = 1, 2, \cdots, n)$ 为数据点 j 属于类 i 的模糊隶属度。

定义 6.6：$d_{ij} = \mu_{ij} \| X_j - V_i \|$ 为 X_j 与类 i 的模糊偏移。

由定义 6.6 可知，$\| \cdot \|$ 为通用的欧氏范式，d_{ij} 为 X_j 与 V_i 之间的欧氏距离。

定义 6.7：$n_i = \sum\limits_x j\mu_{ij}$ 为类 i 的模糊势。

由定义 6.7 可知，$n_i = \sum\limits_x i n_i = n$，在硬划分的极端情况下 n_i 具有确定的数值，即类 i 中的向量个数。

定义 6.8：$\sigma_i = \sum\limits_x j(d_{ij})^2 = (d_{i1})^2 + (d_{i2})^2 + \cdots + (d_{in})^2$ 为类 i 的偏差，即对于任意的类 i 所有数据点偏移的平方和，$\sigma = \sum\limits_x i\sigma_i = \sum\limits_x i \sum\limits_x j\ (d_{ij})^2$ 为所有聚类的总偏差。

由定义 6.8 可知，它们并没有进行标准化处理，并且与所选用的坐标系有关。σ 值越小，则聚类效果越好。

定义 6.9：$\pi = \dfrac{\sigma}{n}$ 为数据集合模糊 c 划分的紧致度，即总偏差与数据集大小的比率。

π 值表示每一聚类的紧致程度，聚类越紧密，则 π 值越小，表明数据集本身的分布特点，与样本数量无关。π 值小，则表明聚类效果较好。Gath 和 Geva 给出了一种模糊超体积的有效性度量(加权总偏差)，可以判断椭圆体聚类以及重叠的聚类。

定义 6.10：$\pi_i = \dfrac{\sigma_i}{n_i}$ 为类 i 的紧致度。

因为类 i 中的向量个数为 n_i，所以 σ_i / n_i 为类 i 的平均偏差。此外，模糊 c 划分的紧致度

还可以定义为 $\pi = \left(\sum_{x} i\pi_i\right)\Big/ c$ 或 $\pi = \max \pi_i$。

定义 6.11：$s=(d_{\min})^2$ 为模糊 c 划分的分离度，其中 $d_{\min}=\min\limits_{i,j}\|V_i - V_j\|$ 为各聚类中心之间的最短距离。

s 值越大，则表明所有的聚类都是分离的。

定义 6.12：$S=\dfrac{\pi}{s}$为基于紧致度和分离度的有效性函数，即：

$$S = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}^2 \ \|V_i - X_j\|^2}{n \min\limits_{i,j} \|V_i - V_j\|^2} \tag{6.63}$$

由定义 6.12 可知，S 的定义与计算 μ_{ij} 所采用的方法无关。S 越小，则表明所有聚类紧密且相互独立。随着 c 的增加而接近 n 时，S 将单调递减。为此，可引入专门的惩罚函数以消除递减趋势。即使不采用惩罚函数，有效性函数 S 仍然是一种行之有效的方法。

由此给出一种采用 S 作为有效性函数的启发策略。对于任何一种模糊聚类算法，发现数据集 X 的一个或多个最佳 c 划分（$c=2,3,\cdots,n-1$）可表示为：

$$\min_{2\leqslant c\leqslant n-1}\{\min_{\Omega_c} S\} \tag{6.64}$$

其中，Ω_c 为对于每个 c 的最适合的候选数据样本。

有效性函数 S 和划分系数 F 都是判断模糊聚类有效性的标准，均可以直接使用，但两者也存在一定的不同之处。首先，F 与任意模糊子集之间的平均重叠程度成反比，而 S 与平均紧致度和分离度成正比；其次，F 缺少与数据集合本身某些特征的直接关联，而 S 则与数据集合的几何特征、距离的度量方法及聚类中心的位置等存在直接关系。

继而又给出了一个新的划分系数 $P(u,c)$，其定义的聚类有效性函数 $\mathrm{FP}(u,c)$为：

$$\mathrm{FP}(u,c) = P(u,c) = \frac{1}{c\sum_{i=1}^{c}\left(\sum_{k=1}^{n} u_{ik}^2 \Big/ \sum_{k=1}^{n} u_{ik}\right)} \tag{6.65}$$

此外，基于聚类本身的物理意义给出了一个模糊聚类有效性函数 $\eta(\chi,c)$。

定义 6.13：给定 $\chi=\{x^p, p=1,2,\cdots,N\}\in R^n$，样本 $x^p=(x_{p1},x_{p2},\cdots,x_{pn})$。设聚类个数为 $c(2\leqslant c<N)$，聚类中心为 $u=(v_1,v_2,\cdots,v_c)^{\mathrm{T}}\in R^n$，则称集合 M_c 是 χ 的一个模糊 c 划分。

$$\begin{aligned} M_c = \{U \in V_{cN} \mid \mu_{pj} \in [0,1], \forall j,p; \sum_{j=1}^{c}\mu_{pj} \\ =1, \forall p; 0 < \sum_{p=1}^{N}\mu_{pj} < N, \forall j\} \end{aligned} \tag{6.66}$$

在式(6.66)中，$\mu_{pj}=\mu_j(x^p)$表示 x^p 属于第 j 类的隶属度，$\mathbf{U}=\{u_1,u_2,\cdots,u_c\}^{\mathrm{T}}$，$u_j=(u_{j1},u_{j2},\cdots,u_{jN})$是样本 χ 的第 j 个聚类集合，V_{cN} 是 $c\times N$ 阶实矩阵的集合。

根据定义 6.13，模糊划分的紧致度定义为 FCM 划分的平均方差，即：

$$\Psi(\chi,c) = \frac{1}{N}\sum_{j-1}^{c}\sum_{p-1}^{N}\mu_{jp}^2 \ (x^p - v_j)^{\mathrm{T}}\mathbf{A}(x^p - v_j) \tag{6.67}$$

其中，$\mathbf{A}$ 为 $n\times n$ 正定加权矩阵，可将欧拉空间扩展到其他空间。

根据定义 6.13,模糊划分的分离度定义为不同聚类中心之间的平均距离,即:

$$\Phi(v) = \frac{1}{|R|}\sum_{j=1}^{c-1}\sum_{h=j+1}^{c}(v_i - v_h)^{\mathrm{T}}\mathbf{A}(v_i - v_h) \tag{6.68}$$

其中,$|R|$为集合 $R=\{(v_i,v_h)|d(v_i,v_h)=\|v_i-v_h\|, i\neq h, 1<i,h\leqslant c\}$的势。

模糊聚类有效性函数 $\eta(\chi,c)$定义为:

$$\eta(\chi,c) = \frac{\Psi(\chi,c)}{\Phi(v)} \tag{6.69}$$

有效性函数 $\eta(\chi,c)$被定义为紧致度和分离度之比。显然,模糊划分形成的同一聚类中的样本靠得越紧,不同聚类中心的距离越远,则模糊聚类结果的合理程度越好。模糊聚类的合理划分即尽量使 $\eta(\chi,c)$越小越好,这样则代表了一个最有效的划分,并由此确定聚类数 c。

但是,当 c 增加而接近 N 时,聚类有效性函数 $\eta(\chi,c)$具有单调性。

聚类有效性是聚类分析的一个瓶颈,对于聚类的成功应用将产生十分深远的影响。

第 7 章　分类和预测

分类(Classification)是数据挖掘的主要功能之一,通过分析类别已知的训练数据,获得描述并区分类别或概念的模型,使用该模型预测并标记未知对象所属的类别。

分类的目的是分析输入数据,通过训练数据表现出来的特性,获得每个类准确的描述或者模型,这种描述常常用谓词表示,使用类描述对未知数据进行分类。尽管这些测试数据的类别是未知的,仍可预测这些新数据所属的类。注意是预测,而不能肯定。

分类的评价遵循以下标准:

- 预测准确率,指模型能够正确预测未知数据类别的能力。
- 速度,指构造和使用模型时的计算效率。
- 鲁棒性,指在数据有噪声或缺失的情况下,模型仍能正确预测的能力。
- 可扩展性,指处理大量数据并构造相应有效模型的能力。
- 易理解性,指所获模型提供的可理解程度。

分类算法是一种有监督的学习算法,其目的是根据训练样本集找出能准确描述并区分类别或概念的模型,以便依据实体的属性值及其他约束条件将其划分到某个类别中。目前,主要的分类算法包括决策树、贝叶斯网络、神经网络、遗传算法、粗糙集和实例推理等。

- 决策树是经典的分类算法,采用自顶向下递归、各个击破的方式构造决策树。树的每一个结点使用信息增益选择属性,从生成的决策树中可提取分类规则。
- KNN(*K*-Nearest Neighbor,*K* 最近邻)是由 Cover 和 Hart 于 1968 年提出,算法主要思路非常简单直观,即如果一个样本在特征空间中的 k 个最相似(特征空间中最邻近)的样本的大多数属于某个类别,则该样本也属于这一类别,即分类时只依据最邻近的一个或几个样本所属的类别决定待分类样本的类别。虽然,KNN 在原理上也依赖于极限定理,但在分类时,只与极少量的相邻样本有关。因此,该方法可以较好地避免样本不平衡的问题。另外,由于 KNN 主要依靠有限的邻近样本,而不是依靠判别类域的方法确定所属类别,因此对于类域交叉或重叠较多的待分类样本集而言,KNN 更为适用。KNN 的不足之处是计算量较大,因为对每一个待分类样本都要计算其到全体样本的距离,才能求得它的 k 个最近邻。目前常用的解决方法是事先对已知样本进行剪辑,去除对分类作用不大的样本。KNN 比较适用于样本数量较大的类域的分类,而那些样本数量较小的类域采用这种算法则较容易产生误分。
- SVM(Support Vector Machine,支持向量机)由 Vapnik 等人于 1995 年提出,具有相对良好的性能。该方法是建立在统计理论基础上的机器学习方法。通过学习 SVM 可以自动寻找出那些对分类有较好区分能力的支持向量,由此构造的分类器可以最大化类与类的间隔,因而具有较好的适应能力和较高的分准率。该方法是由各类域

的边界样本的类别决定最后的分类结果。该算法旨在寻找一个超平面，该超平面可以将训练集中的数据分开，且与类域边界的沿垂直于该超平面方向的距离最大，故SVM亦被称为最大边(Maximum Margin)算法。SVM对小样本量的分类效果较好。

- 贝叶斯是一种已知先验概率与类条件概率的分类方法，待分类样本的分类结果取决于各类域中样本的全体。设训练样本分为 M 类，记为 $C=\{c_1,c_2,\cdots,c_M\}$，每类的先验概率为 $P(c_i)$，其中 $i=1,2,\cdots,M$。当样本量非常大时，可以认为 $P(c_i)=c_i$ 类的样本数/总样本数。对于一个待分类样本 X，其归于 c_i 类的条件概率是 $P(X/c_i)$，根据 Bayes 定理，可得到 c_i 类的后验概率 $P(c_i/X)$ 为：

$$P(c_i/X) = P(X/c_i)P(c_i)/P(X) \tag{7.1}$$

若 $P(c_i/X)=\text{Max}_j P(c_j/X),i=1,2,\cdots,M,j=1,2,\cdots,M$，则 $X\in c_i$ (7.2)

式(7.2)是最大后验概率判决准则，将式(7.1)代入式(7.2)，则若 $P(x/c_i)P(c_i)=\text{Max}_j[P(x/c_j)P(c_j)],i=1,2,\cdots,M,j=1,2,\cdots,M$，则 $X\in c_i$，即常用的贝叶斯分类判决准则。经过长期的研究，理论上贝叶斯论证较充分，应用非常广泛。

贝叶斯分类的不足之处是实际情况下，类别总体的概率分布和各类样本的概率分布函数(或密度函数)常常是未知的，若要获得则要求样本量足够大。另外，贝叶斯分类要求各条件概率相互独立，实际中一般很难满足，因此往往在效果上难以达到理论最大值。

分类和回归都可以实现预测，分类预测的是二元值或离散的类别，而回归预测的是连续或有序值。分类预测和回归预测流程基本相同，即数据准备时，从所有数据集中提取部分数据，作为训练集；同时将剩余的数据作为测试集；利用分类算法对训练集进行分析，得到分类模型；利用测试集对训练后的分类模型进行评估，判断分类模型是否准确。最后，使用模型对未知样本进行预测。

传统的预测方法包括趋势外推法、时间序列法和回归分析法等。

- **趋势外推法**通常用描散点图的方法定性地确定变化趋势，再按照该变化趋势对未来情况做出预测，特点是不对其中的随机成分作统计处理。
- **时间序列法**将因变量(预测目标)和自变量(影响预测目标的因素)均看成随机变量。实际中，多数预测目标的观测值构成的序列表现为(广义)平稳的随机序列或可以转化为平稳的随机序列。虽然在某一给定时刻预测目标的观测值是随机的，但从整个观测序列看，却呈现出某种随机过程(如平稳随机过程)的特性。随机时间序列方法正是依据这一规律建立和估计产生实际序列的随机过程模型，然后利用这些模型进行预测。
- **回归分析法**假定目标同一个或多个独立变量存在相关关系，寻找相关关系的模型。不同于时间序列法的是模型的因变量是随机变量，而自变量是可控变量。回归分为线性和非线性，目前常用多元线性回归模型。

7.1 神经网络

神经网络(Neural Network，NN)是由大量神经元(又称为处理单元)广泛互连组成的网络，是在现代神经生物学基础上提出的模拟生物过程以反映人脑某些特性的计算结构。神

经网络不是人脑神经系统的真实刻画，而只是某种抽象、简化和模拟。信息的处理是由神经元之间的相互作用实现，知识与信息的存储表现为网络元间分布式的物理联系，学习和识别决定于各神经元连接权值的动态演化过程。神经网络是一个高度非线性的超大规模连续时间动力系统。

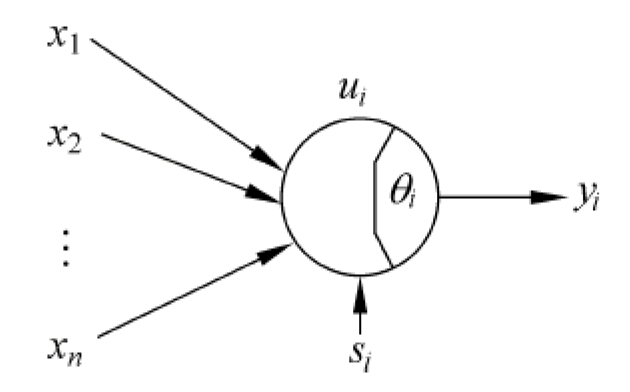


图 7.1　神经元的结构

神经元是神经网络的基本单元，一般是多输入/单输出的非线性器件，其结构和 I/O 特性分别如图 7.1 和图 7.2 所示。

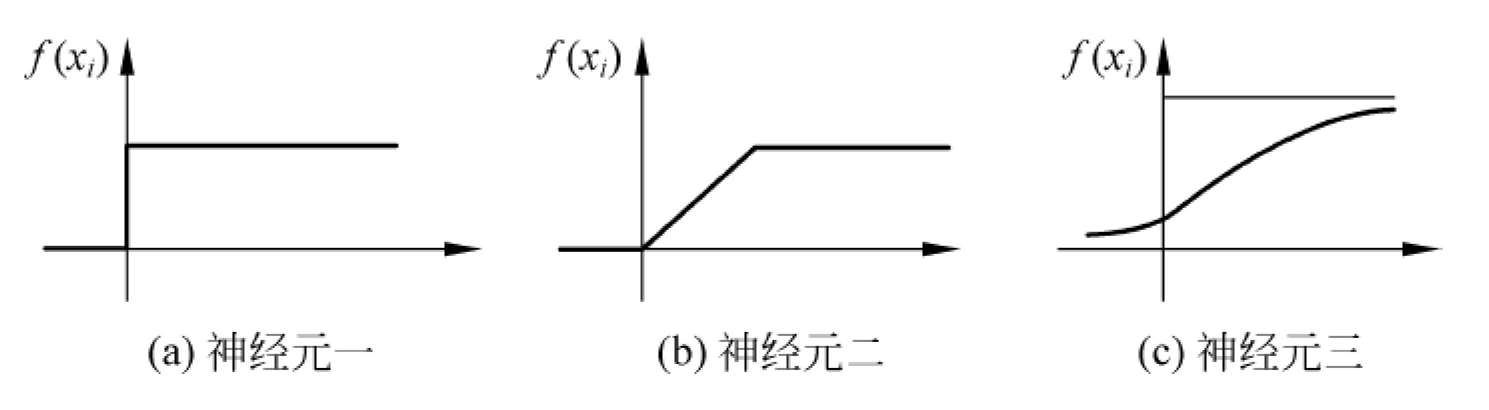


图 7.2　神经元的 I/O 特性

在图 7.1 中，u_i 为神经元的内部状态，θ_i 为域值，x_i 为输入，y_i 为输出，w_{ij} 是神经元 u_i 到 u_j 的连接权值，s_i 为外部输入(某些情况下，它可以控制神经元 u_i 使其保持在某一状态)，神经网络模型可描述为：

$$\begin{aligned} \sigma_i &= \sum_j W_{ji}x_j + s_i - \theta_i \\ u_i &= f(\sigma_i) \\ y_i &= g(u_i) = h(\sigma_i) \qquad h = g \cdot f \end{aligned} \tag{7.3}$$

常用的神经元非线性特性描述如下：

- 域值型，函数 f 为阶跃函数，如图 7.2(a)所示。

$$h(x_i) = f(x_i) = U(x_i) = \begin{cases} 1, x_i > 0 \\ 0, x_i \leqslant 0 \end{cases} \tag{7.4}$$

- 分段线性型，如图 7.2(b)所示。
- S 状，一般没有内部状态并连续取值，其 I/O 特性常用对数或正切等 S 型曲线表示，如 $x_i = \frac{1}{1+\exp(x_i)}$ 或 $x_i = \frac{1}{2}\left[1+\mathrm{th}\left(\frac{\sigma_i}{x_i}\right)\right]$ 等，这类曲线反映了神经元的饱和特性，如图 7.2(c)所示。

目前，已有近四十种神经网络模型，具有代表性的是 BP(Back Propagation，反向传播)、自适应共振理论(Adaptive Resonance Theory，ART)、Hopfield、神经认知机、感知器和自组织映射等。其中，BP 神经网络的应用最为广泛，下面将介绍 BP 神经网络分类器。

BP 神经网络由输入层、隐层和输出层组成，如图 7.3 所示。

BP 神经网络分类是将训练样本的各属性值作为输入，实际类别作为输出。对训练后的 BP 网络，通过剪枝、神经元或活跃值的聚类处理，导出输入层和输出层的关联规则，根据这些规则实现分类。BP 神经网络中，反向传播学习通过一个使目标函数最小化的过程完成输入到输出的映射，目标函数定义为输出层神经元的期望输出与实际输出的误差平方和，即：

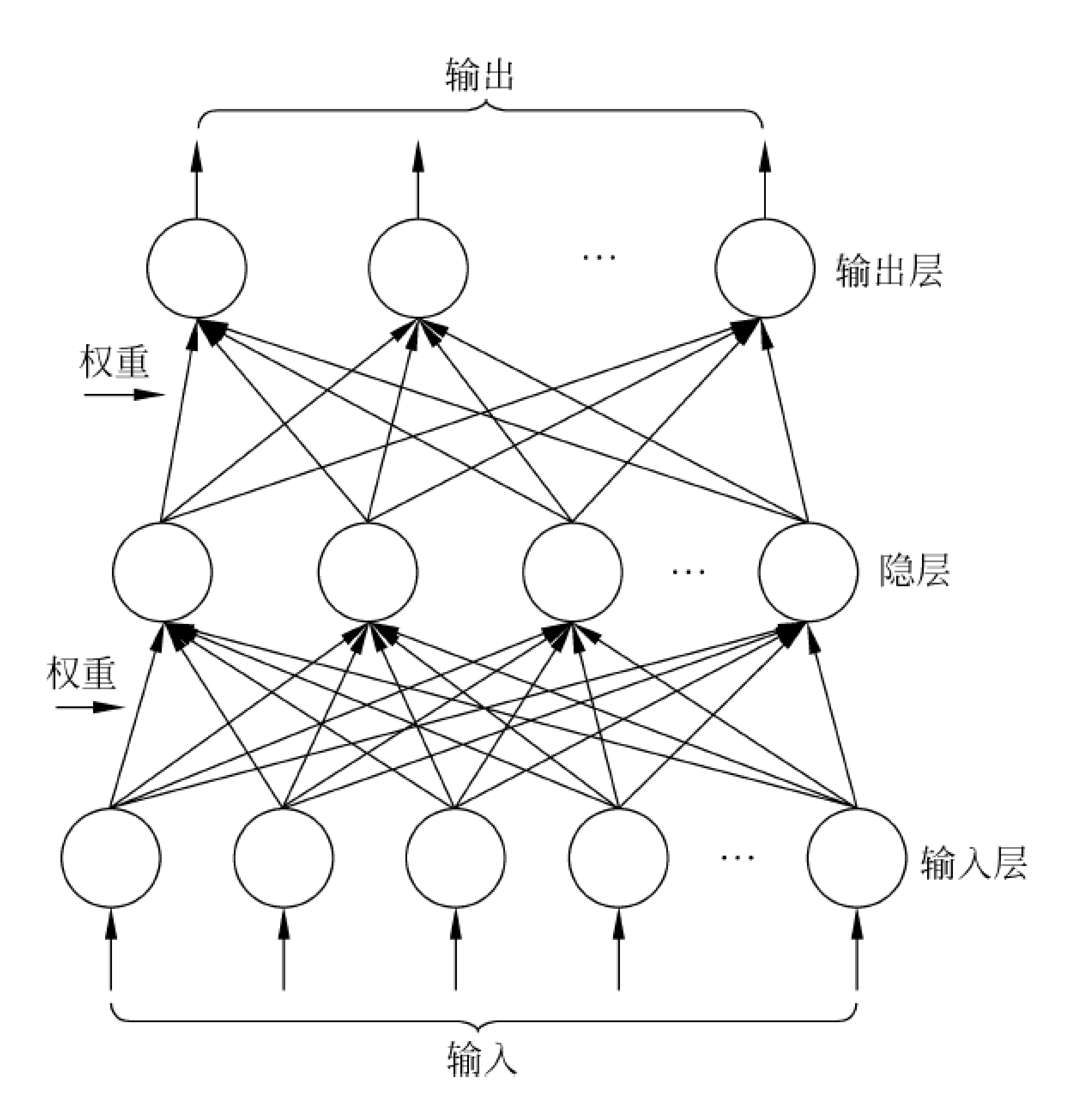


图 7.3 BP 神经网络的结构

$$E = \sum_p E_p = \frac{1}{2}\sum_p \sum_j (t_{pj} - O_{pj})^2 \tag{7.5}$$

其中,t_{pj} 是在输入第 P 个样本时输出层单元 j 实际输出 O_{pj} 的期望输出。目标函数 E 曲面上梯度下降由下面的公式计算得到:

$$-\frac{\partial E}{\partial W_{ji}} = \sum_p \left(-\frac{\partial E_p}{\partial W_{ji}}\right) = \sum_p \delta_{tf} g O_{tf}$$

$$-\frac{\partial E}{\theta_j} = \sum_p \left(-\frac{\partial E_p}{\theta_{ji}}\right) = \sum_p \delta_{tf} \tag{7.6}$$

对于输出层单元 j ,一般化误差 δ_{pj} 表示为:

$$\delta_{tf} = (t_{tf} - O_{tf}) g (1 - O_{pj}) g O_{pj} \tag{7.7}$$

对于隐层,δ_{tf} 表示为:

$$\delta_{tf} = O_{tf} g (1 - O_{tf}) g \sum_p \delta_{tf} W_{tf} \tag{7.8}$$

据此可以调整各连接权值和阈值。

分类前需要确定 BP 神经网络的拓扑结构,即确定隐层的神经元个数及各神经元初始权值和阈值。理论上,隐层的神经元数量越多,逼近越精确。实际上,隐层神经元数量不宜过多,否则会极大地增加训练时间,并造成网络容错能力下降。同时,为了加快学习和训练,需将训练样本各属性值规范化到区间[0,1]。对离散属性可重新编码,使各阈值对应一个输入神经元,例如输入样本 P 为(p_0, p_1,…, p_n),则可设置 I_0,I_1,…,I_n共 $n+1$ 个输入神经元。对输出也需进行规范化,主要针对实际类别与输出神经元的对应关系。当只有两个类别时,可用一个输出神经元表示(神经元输出 0 表示一个类别,1 表示另一个类别);若实际输出多于两个类别,则可以每个类别分别对应一个输出神经元。

BP 神经网络存在的主要问题是：

- 收敛速度。BP 算法最大的弱点是其训练很难收敛，其训练速度非常慢，尤其是当网络训练达到一定的程度。
- 局部极小值。BP 算法采用的是梯度下降法，对一个复杂的网络而言，其误差曲面是一个高维空间的曲面，其中分布着许多局部极小值，一旦陷入局部极小值则很难逃离。
- 网络瘫痪。在训练中，权值可能变得很大，这会使神经元的输入变得更大，导致其激励函数的一阶导数在此点上的值很小。此时的训练步长会变得非常小，最终造成网络停止收敛，即出现所谓的网络瘫痪。

BP 算法的类 C 语言描述如下：

```
输入：训练样本集合，学习率 η，多层前馈神经网络
输出：一个用于对样本分类的 BP 神经网络
初始化 BP network 的权值 Wij 和阈值 θj；
   while 不满足训练终止条件
   {
     for samples 中的各训练样本 X                  //正向传播输入
     {
       for 隐层或输出层的各神经元 j
       {
           Ij = ∑iWij * Oi + θj;                    //相对于前一层 i，计算神经元 j 的输入 Ij
           Oj = 1/(1 + e- Ij);                      //使用对数型的单极性 Sigmoid 函数将各神
                                                    //经元 j 的输出映射到区间[0,1]
         for 输出层的各神经元 j                     //反向传播误差
         ERRj = Oj * (1 - Oj) * (Tj - Oj);          //根据训练样本的已知类标号真实输出 Tj，
                                                    //计算神经元 j 的误差 ERRj
         for 从最后 1 个到第 1 个隐层的各神经元 j
         ERRj = Oj * (1 - Oj) * ∑k(ERRk * Wjk);     //根据下一较高层中连接到 j 的所有神经元
                                                    //的误差加权值来计算隐层神经元 j 的误
                                                    //差 ERRj
         for BP network 中的各权值 Wij              //更新权值
           {
             ΔWij = η * ERRj * Oj                   //实际应用时，η 一般经验性地设置为训练样
                                                    //本集迭代次数的倒数
             Wij = Wij + ΔWij
           }
         for BP network 中的各阈值 θj               //更新阈值
           {
             Δθj = η * ERRj
             θj = θj + Δθj
           }
       }
     }
   }
```

BP 算法的流程如图 7.4 所示。

其学习过程分为正向传播输入和反向传播误差。样本从输入层经隐层到输出层，逐层处理时，各层神经元状态只对下一层神经元的状态产生影响。在输出层，实际输出和期望输出不一致时，进入反向传播过程，即将误差反向传播，并通过更新权值和阈值调整，使误差趋向最小。

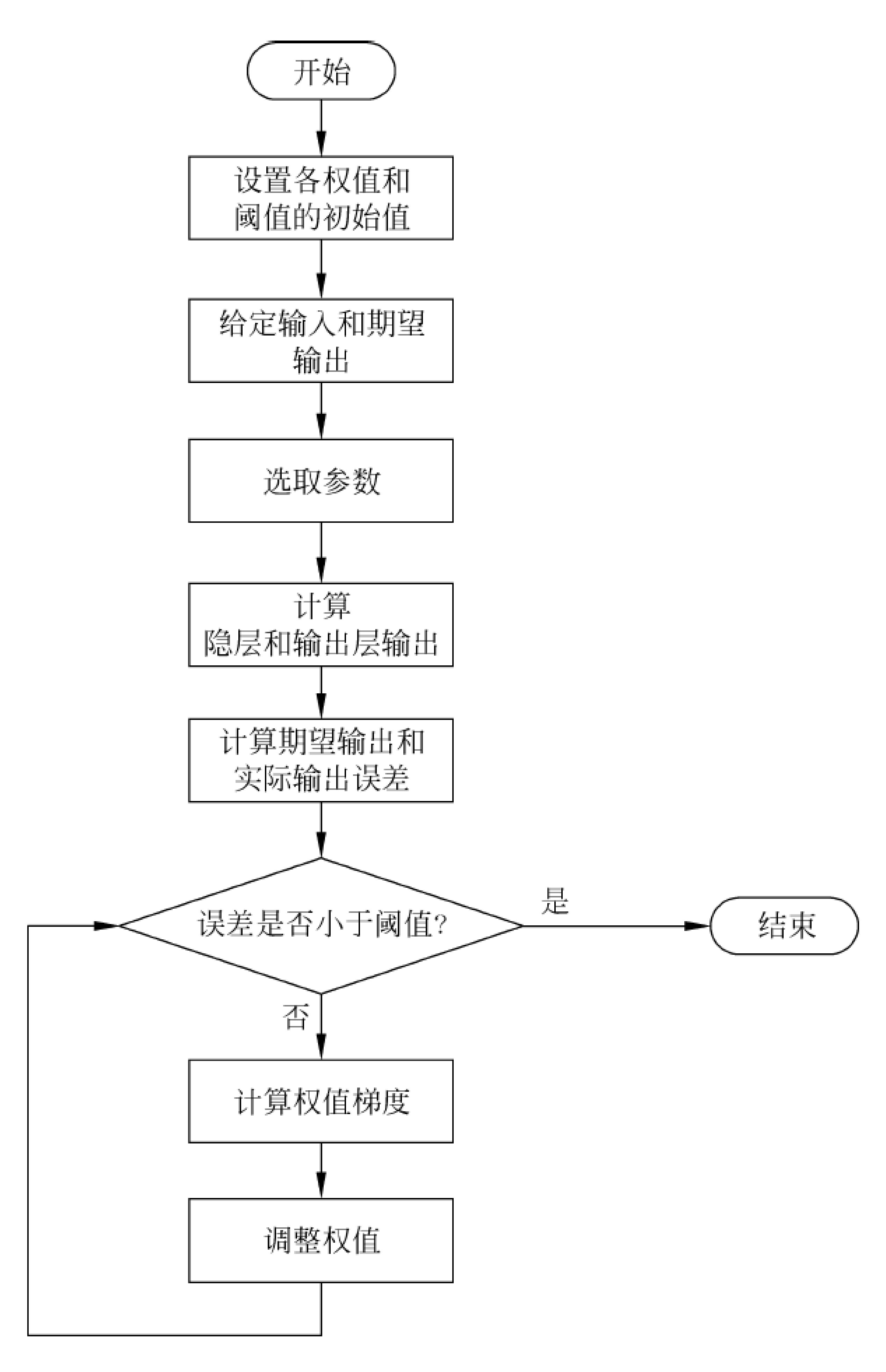


图 7.4 BP 算法的流程图

7.2 决策树

决策树(Decision Tree)是分类预测的主要方法,采用基于实例的归纳学习,旨在从一组无序、无规则的实例中推理出决策树形式的分类规则,采用自顶向下的递归方式,在决策树的内部结点进行属性值的比较并根据不同属性判断从该结点向下的分支,在决策树的叶结点得到结论,所以从根到叶结点对应一条合取规则,整棵树对应一组析取规则。

决策树分类是利用属性值对各子集逐级划分,直到一个结点仅含有同一类样本为止。最早起源于 Hunt 等人提出的概念学习系统(Concept Learning System,CLS),然后发展到 Quinlan 的 ID3 算法,最后演化为能处理连续值属性的 C4.5 算法。

1. 主要算法

下面主要介绍三种。

1) CLS

以一棵空决策树开始,通过增加结点逐步求精,直到产生一棵能正确分类训练样本的决

策树为止，是一个循环递归过程。假设 T 为已知的训练集，则：

（1）如果 T 的所有样本均为正例，则生成一个 YES 结点并终止；如果 T 的所有样本均为反例，则生成一个 NO 结点并终止；否则，根据某种启发式策略选择一个属性 A，设 A 取值为 $v_1, v_2 \cdots v_r$，并生成新结点。

（2）将 T 根据其属性 A 的取值进行划分，生成 r 个子集记为 $T_1, T_2, \cdots, T_r$。

（3）递归地应用该算法到每个子集 T_i。

CLS 中，分类属性的选择决定了算法的效率与所生成决策树的繁简程度和预测效果。属性选择是决策树归纳算法的关键。

CLS 可以产生所有可能的决策树，正确分类训练样本，并能选择最简单的决策树。但是属性选择范围在实际应用中往往受到问题大小的限制。

2）ID3

Quinlan 提出著名的 ID3 算法，是对 CLS 的改进，通过选择窗口形成决策树，利用信息论中的互信息或信息增益选择具有最大信息量的属性，建立决策树的一个结点，再根据该属性的不同取值建立树的分支，在每个分支中重复建立树的下层结点和分支，效果非常理想。其优点是描述简单，分类速度快，特别适合大规模数据集。但算法引用信息论中的互信息作为单一属性能力的度量，试图减少树的平均深度，忽略了叶结点数目的研究，其启发式函数并不是最优的，存在的主要问题是：

（1）互信息的计算依赖于取值数目较多的属性，而取值较多的属性不一定最优。

（2）ID3 是非递增学习算法。

（3）抗噪能力差，训练集中正例和反例较难控制。改进算法有 C4.5 以及 CART（引进可调错误率概念）等。

下面给出 ID3 算法的一个实例。例如根据天气状况预测某天是否适合打高尔夫球，适合的属于正例记为 P，不适合的属于反例记为 N。天气由四种属性描述，即 Outlook 取值分别为 sunny、overcast 和 rain；Temperature 取值分别为 cool、mild 和 hot；Humidity 取值为 normal 和 high；Windy 取值为 false 和 true。训练集共 14 个样本，如表 7.1 所示。

表 7.1　打高尔夫球天气形势的训练集

序号	Outlook	Temperature	Humidity	Windy	Class
1	sunny	hot	high	false	N
2	sunny	hot	high	true	N
3	overcast	hot	high	false	P
4	rain	wind	high	false	P
5	rain	cool	normal	false	P
6	rain	cool	normal	true	N
7	overcast	cool	normal	true	P
8	sunny	mild	high	false	N
9	sunny	cool	normal	false	P
10	rain	mild	normal	false	P
11	sunny	mild	normal	true	P
12	overcast	mild	high	true	P
13	overcast	hot	normal	false	P
14	rain	mild	high	true	N

由表 7.1 可知，$P=9$，$N=5$，则

$$I(P,N)=-\frac{9}{14}\log_2\frac{9}{14}-\frac{5}{14}\log_2\frac{5}{14}=0.940$$

$$\begin{aligned}E(\text{outlook})&=\frac{5}{14}I(P_1,N_1)+\frac{4}{14}I(P_2,N_2)+\frac{5}{14}I(P_3,N_3)\\&=\frac{5}{14}I(2,3)+\frac{4}{14}I(4,0)+\frac{5}{14}I(3,2)\\&=\frac{5}{14}\times 0.971+\frac{4}{14}\times 0+\frac{5}{14}\times 0.971\\&=0.694\end{aligned}$$

$$\text{Gain}(\text{Outlook})=I(P,N)-E(\text{outlook})=0.940-0.694=0.246$$

类似地，可得：

$$\text{Gain}(\text{Temperature})=0.029$$

$$\text{Gain}(\text{Humidity})=0.151$$

$$\text{Gain}(\text{Windy})=0.048$$

显然 Outlook 的信息增益最大，因此 Outlook 首先被选为根结点并向下扩展，以此类推，得到相应的 ID3 决策树，如图 7.5 所示。

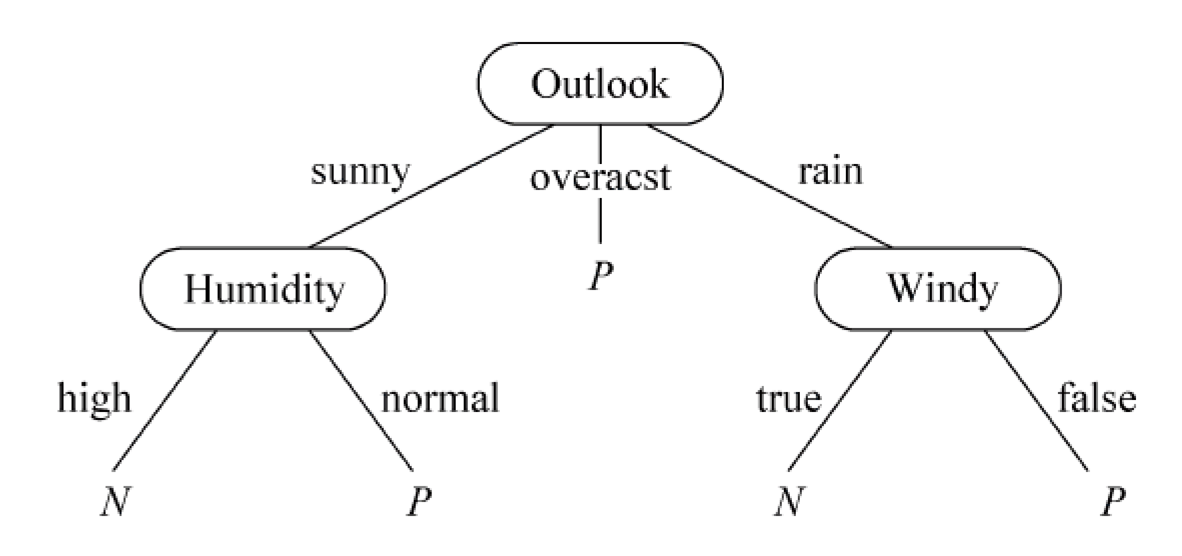


图 7.5　打高尔夫球的 ID3 决策树

3) C4.5

ID3 有很多改进算法，其中 Quinlan 在 1994 年开发的 C4.5 流行最广。C4.5 的改进主要体现在两个方面：

(1) 解决连续取值的学习问题。

(2) 提供学习结果决策树到等价规则集的转换功能。

C4.5 属于一种归纳学习(Inductive Learning)算法，旨在从大量的训练样本中归纳抽取一般的判定规则和模式，是机器学习(Machine Learning)领域最成熟的分支之一。根据有无监督指导，归纳学习又分为有监督学习(Supervised Learning)和无监督学习(Unsupervised Learning)。有监督学习可分为覆盖算法(Covering Algorithm)和分治算法(Divide-and-conquer Algorithm)，前者归纳生成规则，后者归纳生成决策树。C4.5 属于有监督的学习算法。

4) SLIQ 算法

构造决策树时，SLIQ 采用预排序和广度优先策略。在一般决策树中，使用信息增益评价结点分裂的质量。SLIQ 使用 Gini 指标(Gini Index)代替信息增益，对有 n 个分类的数据集 S，gini(S)定义为：

$$\mathrm{gini}(S) = 1 - \sum p_j \times p_j \tag{7.9}$$

其中，p_j 是 S 中第 j 类数据的频率。gini 越小，信息增益(Information Gain)越大。

区别于一般的决策树，SLIQ 采用二分查找树结构。对每个结点都需要先计算最佳分裂方案，然后执行分裂。对于连续值属性(Numerical Attribute)分裂可以先对属性值排序，假设排序后的结果为 $V_1, V_2, \cdots, V_n$，因为分裂只发生在两个结点之间，所以有 $n-1$ 种可能性。从小到大依次取不同的分裂点，通常取中点 $(V_i+V_{i+1})/2$ 作为分裂点，使信息增益最大(即 gini 最小)的即最佳分裂。因为每个结点都需要排序，所以代价很大，降低排序成本成为一个重要的问题，SLIQ 对此有很好的解决方案。对于离散值属性(Categorical Attribute)，设 $S(A)$ 为 A 的所有可能的值，属性分裂将遍取 S 的所有子集 S'，获得分裂成 S' 和 $S-S'$ 时的 gini 指标。当 gini 最小时，即最佳分裂。显然，这是一个遍历集合 S 的所有子集的过程，共需要计算 2 次，代价很大。SLIQ 对此也有一定程度的优化。

SLIQ 的处理能力比 ID3 和 C4.5 强大得多，因此在一定程度上具有良好的可扩展性，但仍存在不足之处，主要体现在：

(1) 需要将类别列表存放于内存，而类别列表的长度与训练集的长度相同，这在一定程度上限制了处理数据集的大小。

(2) 进行预排序，而其复杂度本身并不是与样本数量成线性关系，因此 SLIQ 不可能达到随样本数增长的线性可扩展性。

2. 算法描述

ID3 算法的基本思路是首先在数据集中用信息增益作为属性选择的标准找出最有影响力的属性，将数据集分成多个子集，每个子集再选择最具影响力的属性进行划分，一直进行到所有子集仅包含同一类型的数据为止，最后得到一棵决策树。决策树的构造采用自上而下、分而治之的递归方式。初始时，根结点包含数据集中的所有样本。若一个结点包含的样本均为同一类别，则该结点成为叶结点并标记为该类别。否则，采用信息增益的度量选择合适的分类属性，将数据集划分为若干个子集。该属性称为相应结点的测试属性(Test Attribute)。对测试属性的每个已知值都创建一个分支，同时也包含一个被划分的子集。递归地对所获得的每个划分形成一个决策子树。一旦一个属性出现在某一个结点上，则不能再出现在该结点之后所产生的子树结点上。当一个结点包含的所有样本均为同一类别或没有样本满足测试属性值，则算法终止。

基于信息增益选择测试属性的方法，使得对一个对象分类所需的期望测试数达到最小，确保得到一棵简单的树。在该方法中选择具有最高信息增益的属性作为当前结点的测试属性，该属性使得数据样本分类所需的信息量最小，并反映划分的最小随机性或“不纯性”。属性信息增益的计算方法如下：

设数据集合 S 有 s 个样本，类别属性有 m 个不同的取值，定义 m 个不同的类 $C_i, i \in \{1, 2, 3, \cdots, m\}$。设 s_i 为类别 C_i 的样本个数，则对一个数据集分类所需要的期望信息为：

$$I(s_{1j}, s_{2j}, \cdots, s_{mj}) = -\sum_{i=1}^{m} p_{ij} \log_2(p_{ij}) \tag{7.10}$$

其中 p_i 是任意一个对象属于类别 C_i 的概率，可以按 s_i/S 计算。因为采用二进制编码，所以对数函数以 2 为底。

设属性 A 可取 v 个不同的值 $\{a_1, a_2, \cdots, a_v\}$。可以用属性 A 将 S 划分为 v 个子集 $\{S_1,$

$S_2,\cdots,S_v\}$,其中 S_j 包含 S 中属性 A 取值为 a_j 的样本。若属性 A 为测试属性,设 s_{ij} 为子集 S_j 中属于 C_i 类别的样本数。则利用属性 A 划分当前集合所需要的期望信息计算如下:

$$E(A)=\sum_{i=1}^{v}\frac{s_{1j}+s_{2j}+\cdots+s_{mj}}{S}I(s_{1j},s_{2j},\cdots,s_{mj}) \tag{7.11}$$

其中,$\frac{s_{1j}+s_{2j}+\cdots+s_{mj}}{S}$称为第 j 个子集的权值。$E(A)$值越小,表示子集划分结果越好。而对于一个给定子集 S_j,其期望信息如式(7.10)所示,其中 $p_{ij}=\frac{s_{ij}}{|S_j|}$为子集 S_j 中任一个样本属于类别 C_i 的概率。

由此利用属性 A 对当前分支结点进行划分所获得的信息增益是:

$$\text{Gain}(A)=I(s_{1j},s_{2j},\cdots,s_{mj})-E(A) \tag{7.12}$$

Gain(A)是根据属性 A 进行集合划分所获得的信息熵的减少量。

ID3 计算每个属性的信息增益,从中选择信息增益最大的属性作为给定集合的测试属性并由此产生相应的分支结点,所产生的结点被标记为相应的属性,并根据这一属性的不同取值分别产生相应的决策树分支,每个分支代表一个被划分的子集。

ID3 算法简单,学习能力较强,但仅对较小的数据集有效,且对噪声比较敏感,当数据集增大时,决策树可能会改变。C4.5 继承了 ID3 的优点,同时对 ID3 进行了改进,例如能够完成对连续属性的离散化处理,能够对不完整数据进行处理等。

决策树的生成分为两个步骤:

(1) 数据从根结点开始递归地进行数据分片。

下面以 C4.5 为说明决策树生成算法:

```
Procedure C4.5BuildTree (S,A )
//S: 训练样本集, A: 分类属性集合
if  属性是连续的则进行散化处理
if  所有样本属于同一分类,返回标号为该分类的叶结点
else if  属性值为空,返回标号为 S 最普遍分类的叶结点
    else{
        For 每一个属性 A
        估计该结点在 A 上的信息增益选出最佳的属性 A,将 S 分裂为 Sᵢ,长出分支
        (Sᵢ 为属性 Aⱼ的第 i 个值对应的样本集)
        For each Sᵢ
        if Sᵢ 为空 then 返回叶结点,标记为 S 中最普遍的类
        else  C4.5BuildTree (Sᵢ,A - Aⱼ)
            }
```

(2) 通过修剪去掉一些可能是噪声或者异常的数据。

决策树分类算法的输入是一组带有类别标记的训练样本;输出是一棵二叉或多叉树。二叉树的内部结点(非叶子结点)一般表示为一个逻辑判断,例如形为 $a_i\Rightarrow v_i$ 的逻辑判断,其中 a_i 是属性,v_i 是该属性的某一取值;树的边是逻辑判断的分支结果。多叉树(如 ID3)的内部结点是属性,边是该属性的所有取值,有几个属性值,就有几条边。树的叶子结点均为类别标记。决策树的构造采用自上而下的递归方法。以多叉树为例,构造过程是如果训练样本集合的所有样本是相同类别的,则将其作为叶子结点,结点标记为该类别。否则,根据某种策略选择一个属性,按照属性的各个取值,把集合划分为若干子集,使得每个子集的

所有样本在该属性上具有同样的属性值。然后再依次递归处理各个子集。本质上是“分而治之”(Divide and Conquer)。二叉树同理,差别仅在于要选择一个好的逻辑判断。构造决策树的关键是如何选择恰当的逻辑判断或属性。对于同样一组样本,可以有很多决策树能符合这组例子。研究表明,一般情况下,树越小则预测能力越强。应构造尽可能小的决策树,关键在于选择恰当的逻辑判断或属性。由于构造最小的树是NP问题,因此只能采用启发式策略选择逻辑判断或属性,如信息增益、信息增益比(Gain Ratio)、Gini Index和正交法等,不同的度量效果不同,特别是对于多值的属性。C4.5算法使用信息增益作为启发策略构造决策树。

实际中,用于分类模型学习的训练样本往往不是完美的,原因在于:

- 某些属性字段上缺值(Missing Value)。
- 缺少必须的数据而造成数据不完整。
- 数据不准确含有噪声甚至是错误的。

因此,克服噪声需要对决策树剪枝。

剪枝旨在克服噪声,同时也能简化树使之更易于理解。剪枝类型主要包括:

- 向前剪枝(Forward Pruning)在生成树的同时决定是继续对不纯的训练子集进行划分还是停机。
- 向后剪枝(Backward Pruning)是一种两阶段法,即拟合-化简(Fitting and Simplifying),首先生成与训练样本完全拟合的一棵决策树,然后从树的叶子开始剪枝,逐步向根的方向剪。

但是,剪枝也存在一定局限性。剪枝并不是对所有的数据集都适用,就像最小的树并不是最好(具有最大的预测率)的树。当数据稀疏时,应防止过分剪枝(Overpruning)。从某种意义而言,剪枝也是一种偏向,对有些数据集效果好,而有些数据集则效果较差。

7.3 实现过程

分类的实现过程一般分为两个步骤。

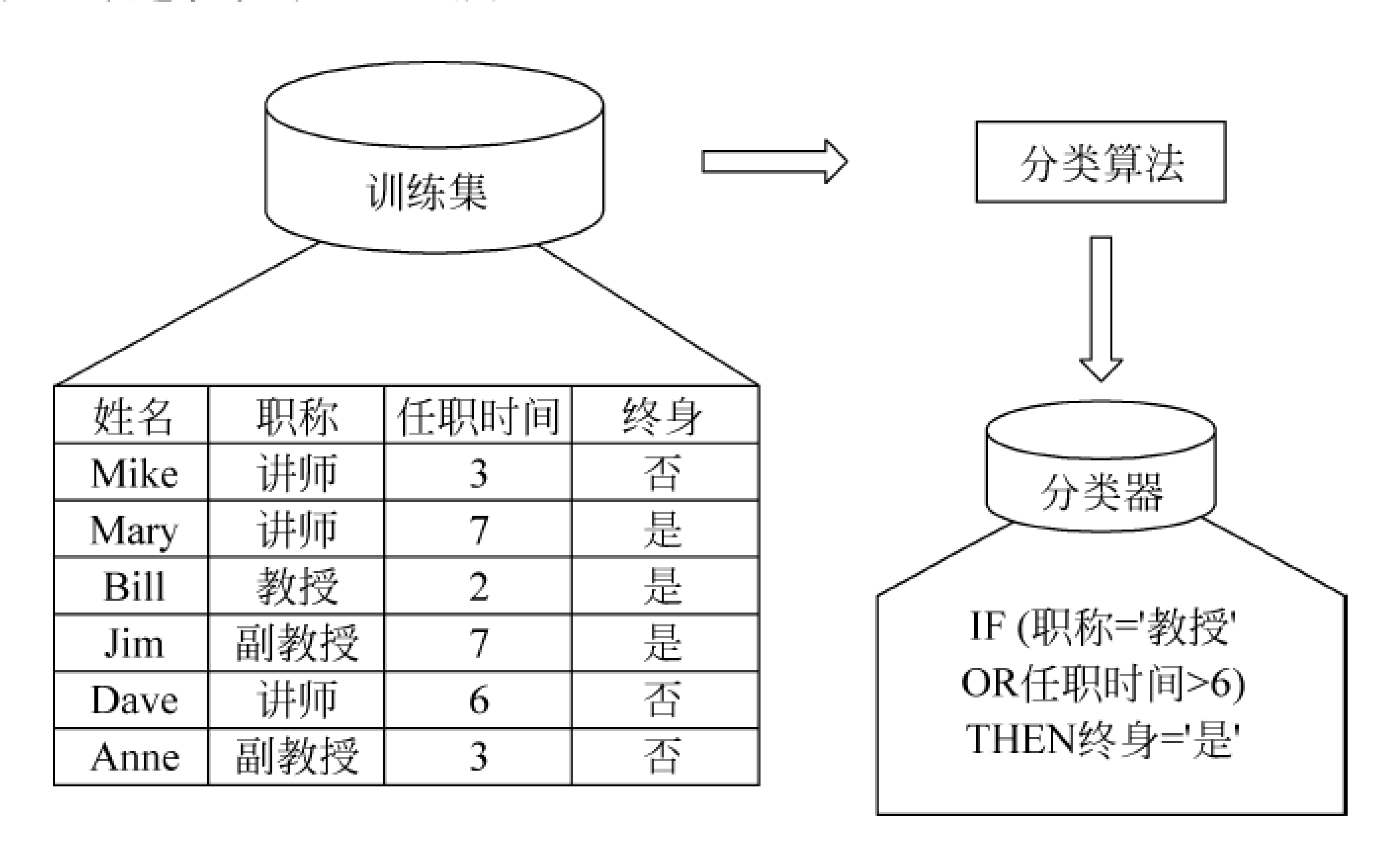
1. 通过训练集建立模型,即构建模型

构建模型一般分为训练和测试两个阶段。构造模型之前,要求将数据集划分为训练集和测试集,并对每个样本进行类别标记,即预设分类类别。训练阶段,通过分析由属性描述的训练样本构造模型。该阶段也称为有指导的学习,通常模型为分类规则、判定树或数学公式的形式;测试阶段,使用测试集评估模型的分类准确率,如果认为模型的准确率可以接受,则可以使用该模型对类别未知的待分类样本进行分类。一般而言,测试阶段的代价远低于训练阶段。

为了提高分类的准确性、有效性和伸缩性,构建模型前通常需要预处理,具体包括:

- 数据清理。其目的是消除或减少噪声,处理缺失值。
- 相关性分析。由于数据集的许多属性可能与分类不相关,若包含这些属性将减慢或可能误导学习。相关性分析的目的是删除不相关或冗余的属性。
- 数据变换。数据可以抽象到较高层次。例如,属性“收入”的连续值可以概化为离散值,如低、中和高。此外,还可以规范化,即将属性值按比例缩放到较小的区间,如[0,1]。

构建分类模型的过程如图 7.6 所示。

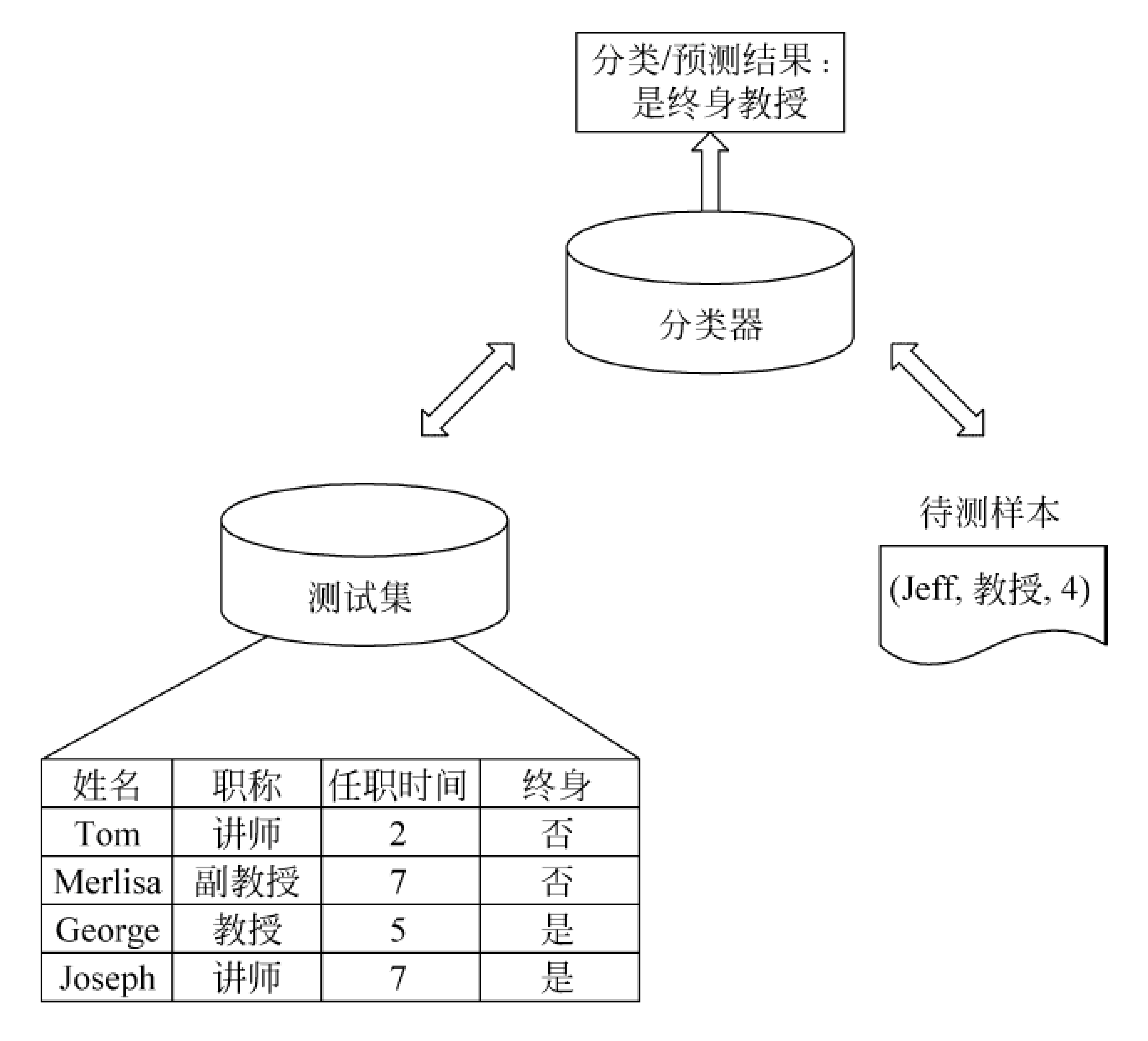


姓名	职称	任职时间	终身
Mike	讲师	3	否
Mary	讲师	7	是
Bill	教授	2	是
Jim	副教授	7	是
Dave	讲师	6	否
Anne	副教授	3	否

图 7.6 构建分类模型的过程

2. 利用已训练好的模型识别类型未知的对象,即使用模型

测试集与训练集相分离,否则将出现过度拟合(Overfitting)现象。使用分类模型的过程如图 7.7 所示。



姓名	职称	任职时间	终身
Tom	讲师	2	否
Merlisa	副教授	7	否
George	教授	5	是
Joseph	讲师	7	是

图 7.7 使用分类模型的过程

第8章 关联分析

8.1 概述

自然界中某一事物发生时其他事物也会发生,这样一种联系称为关联。反映事件之间依赖或关联的知识称为关联型知识(又称依赖关系)。关联是指两个或多个变量取值之间存在的某种重要的可被发现的规律性,分为简单关联、时序关联和因果关联等。

以交易型数据库为例,一个交易一般由交易处理时间,一组顾客购买的物品,有时也包括顾客标识(如信用卡号)等组成。关联规则可描述在一次交易中物品之间同时出现的规律。更确切地说,关联规则量化地描述了物品 X 出现对物品 Y 出现有多大的影响。

例如,体育用品商店通过对销售数据的关联分析发现这些数据中常常隐含这样的规律,即"购买篮球的顾客中有70%的人同时购买运动服,所有交易中有40%的人同时购买篮球和运动服"等,这些规律即关联规则。

另一个典型例子是购物篮的分析。通过发现顾客放入其购物篮中不同商品之间的联系,进一步分析顾客的购买习惯,帮助零售商制定针对性的营销策略,如合理地安排货架以引导销售,将牛奶和面包尽可能放近一些,以刺激一次购物同时购买多种商品。

频繁项集(Frequent Itemset)是产生关联规则的基础,因此在定义关联规则前首先介绍频繁项集的定义及其性质。

定义 8.1:设 I 为一个由 m 个项组成的集合 $I=\{i_1,i_2,\cdots,i_m\}$ 称为项集(Itemset)。

交易 T 为由 I 中的项组成的子集,即 $T\subset I$。与集合的定义一样,交易 T 中同样不存在重复的元素。假设这里所涉及的交易和项集的项都已排序。

如果交易 T 包含项集 X 中的所有项,即 $X\subset T$,则称 T 支持 X。$T(X)$ 定义为所有支持 X 的交易组成的集合。

定义 8.2:数据库 D 中支持项集 X 的交易所占的比例称为 X 在 D 中的支持度,记为 $\mathrm{supp}(X)$ 或 $S(X)$。

设 minsup 为给定的最小支持度,如果 $\mathrm{supp}(X)\geqslant \mathrm{minsup}$,则称项集 X 是频繁的。一个项集的最小支持度是该项集被认为是有意义的,支持它的交易占数据库 D 中所有交易总和的最小比例,通常是根据经验设定,不具有最小支持度的项集被认为是没有意义的。

定义 8.3:项集包含项的个数称为长度或基数。长度为 k 的项集称为 $k(k=|X|)$ 维项集,记为 k-itemset。

频繁项集具有以下三个性质,其中性质8.1和性质8.3是所有关联分析算法的基础。

性质 8.1:子集支持

设 A 和 B 是两个不同的项集,如果 $A\subset B$,则 $\mathrm{supp}(A)\geqslant \mathrm{supp}(B)$。因为所有支持 B 的交易也一定支持 A。

性质 8.2：非频繁项集的超集也一定是非频繁的。

如果 A 不满足最小支持度条件，即 $\text{supp}(A)<\text{minsup}$，则 A 的每个超集 B 也不是频繁的。由性质 8.1 可得 $\text{supp}(B)\leqslant\text{supp}(A)\leqslant\text{minsup}$，因此 B 也是非频繁的。

性质 8.3：频繁项集的子集也是频繁的。

如果项集 B 是数据库 D 中的频繁项集，即 $\text{supp}(B)\geqslant\text{minsup}$，则 B 的每个子集 A 也是频繁的。由性质 8.1 可得 $\text{supp}(A)\geqslant\text{supp}(B)\geqslant\text{minsup}$，因此 A 也是频繁的。特别地，如果 A 是频繁的，则它的 k 个基数为 $k-1$ 的子集都是频繁的；反之则不成立。

设 $I=\{i_1,i_2,\cdots,i_m\}$ 是有 m 个不同元素的集合，T 是针对 I 的交易集合，每一笔交易包含若干个属于 I 的项。关联规则表示为 $X\Rightarrow Y$，其中 X 和 Y 是两个不相交的集合，$X,Y\subset I$，并且 $X\cap Y=\varnothing$。X 称为规则的前提或前项，Y 称为规则的结果或后项。每个规则都有两个度量，即支持度(Support)和可信度(Confidence)。其中，

支持度定义为 $\text{support}\ (X\Rightarrow Y)\ =\text{support}\ (X\cup Y)$。

可信度定义为 $\text{confidence}(X\Rightarrow Y)=\text{support}(X\cup Y)/\text{support}\ (X)$。

关联规则的形式为 R：$X\Rightarrow Y$。

关联规则具有以下四个性质。

性质 8.4：规则的非结合性

如果关联规则 $X\Rightarrow Z$ 和 $Y\Rightarrow Z$ 在 D 中成立，规则 $X\cup Y\Rightarrow Z$ 不一定在 D 中成立。如果 $X\cap Y=\varnothing$，并且 D 中支持 Z 的所有交易都只支持 X 或 Y，则集合 $X\cup Y\cup Z$ 的支持度为 0，因此 $X\cup Y\Rightarrow Z$ 的可信度为 0。

类似地，如果规则 $X\Rightarrow Y$ 和 $X\Rightarrow Z$ 成立，规则 $X\Rightarrow Y\cup Z$ 不一定成立。

性质 8.5：规则的不可分解性

如果关联规则 $X\cup Y\Rightarrow Z$ 在 D 中成立，规则 $X\Rightarrow Z$ 和 $Y\Rightarrow Z$ 不一定在 D 中成立。例如，当 Z 只出现在一个交易中时，如果 X 和 Y 也出现在其中，即 $\text{supp}\ (X\cup Y)\ =\text{supp}(Z)$，规则就是不可分解的。另外，如果 X 与 Y 的支持度与 XY 的支持度相比足够大，就会使得分解后的两个规则不具有所要求的可信度，因而规则也不可分解。

因为 $\text{supp}(X\cup Y)\geqslant\text{supp}(X\cup Y\cup Z)$ 且 $\text{supp}(X\cup Z)\geqslant\text{supp}(X\cup Y\cup Z)$，所以如果规则 $X\Rightarrow Y\cup Z$ 成立，则规则 $X\Rightarrow Y$ 和 $X\Rightarrow Z$ 都成立。由此可得较小规则的支持度与可信度与原规则相比都有所增加。

性质 8.6：规则的不可传递性

由关联规则 $X\Rightarrow Y$ 和 $Y\Rightarrow Z$ 成立不能推出规则 $X\Rightarrow Z$ 成立。设 $T(X)\subset T(Y)\subset T(Z)$，最小可信度为 minconf，$\text{conf}(X\Rightarrow Y)=\text{conf}(Y\Rightarrow Z)=\text{minconf}$。

由 $T(X)\subset T(Y)$，可得 $\text{conf}\ (X\Rightarrow Y)=S(X\cup Y)/S(Y)$

$$=S(X)/S(Y)=\text{minconf}$$

由 $T(Y)\subset T(Z)$，可得 $\text{conf}(Y\Rightarrow Z)=S(Y\cup Z)/S(Z)$

$$=S(Y)/\ S(Z)=\text{minconf}$$

由 $\text{T}(X)=\text{T}(Z)$，可得 $\text{conf}(X\Rightarrow Z)=S(X\cup Z)/S(Z)=S(X)/S(Z)$

由上面三个等式及 $\text{minconf}<1$，可得 $\text{conf}\ (X\Rightarrow Z)=\text{minconf}\ <\text{minconf}$。因此规则 $X\Rightarrow Z$ 不成立。

性质 8.7：规则的可扩展性

设项集 L、A 和 B，且 $B \subset A \subset L$，如果规则 $A \Rightarrow (L-A)$ 不满足最小可信度条件，则 $B \Rightarrow (L-B)$ 也不满足最小可信度条件。由性质 8.1 可得 $\text{supp}(B) \geqslant \text{supp}(A)$，再由可信度的定义可得 $\text{conf}(B \Rightarrow (L-B)) = \text{supp}(L)/\text{supp}(B) \leqslant \text{supp}(L)/\text{supp}(A) < \text{minconf}$。

同理可得，对项集 L、D 和 C，且 $D \subset C \subset L$，$D=0$，如果规则 $(L-C) \Rightarrow C$ 成立，则规则 $(L-D) \Rightarrow D$ 也成立。

当所有频繁项集及其支持度确定后，利用性质 8.7 可以加速规则的产生。

关联分析旨在发现支持度或可信度分别大于设定的最小支持度值和最小可信度值的规则。本质上，关联分析分解成：

- 产生所有支持度大于或等于指定最小支持度的项集，这些项集称为频繁项集，而其他的项集则称为非频繁项集。
- 对于每个频繁项集，产生可信度大于或等于最小可信度的规则，即对于一个频繁项集 L 及任意 $S \subset L$，如果 $\text{support}(L)/\text{Support}(S) \geqslant \text{minconf}$，则规则 $S \Rightarrow (L-S)$ 就是一个正确规则。

关联分析的主要算法包括 R. Agrawal 等提出的 AIS、Apriori 及其变种 AprioriTid 和 AprioriHybrid，M. Houtsma 等提出的 SETM，J. Park 等提出的 DHP，A. Savasere 等提出的 PARTITION，H. Toivonen 提出的 Sampling 和 Jiawei Han 提出的 FP-Growth 等。

AIS 算法的主要思想是在扫描数据库的同时产生候选项集并累计支持度。具体地，在对数据库进行第 k 次扫描时，候选项集是由第 $k-1$ 次扫描所产生的边界集通过增加当前事务中的项得到，同时计算候选项集元素的支持度，直到某次扫描所产生的边界集为空。主要缺点是生成的候选项集过大。

SETM 算法实际上是 AIS 的变种，把候选集的产生和累计分开。在一个线性存储结构中存储了所有候选集和相应交易的标识符(TID)。每次扫描结束后，不再读取数据库，而是对 TID 进行排序并累计各个候选集的支持度。其主要思想是通过扫描候选项集的编码代替扫描数据库，实质上是把数据库中与支持度有关的信息单独提取出来，构成一个较小但充分的 TID 库，这大大减少了数据库访问时间，不足之处同样是候选项集过大。

Apriori 算法利用项集的性质对数据库进行多次扫描，即任意频繁项集的子集都是频繁项集；任意非频繁项集的超集都是非频繁项集。第一次扫描数据库得到频繁 1 项集 L_1，第 k 次扫描前先利用上次扫描结果，即频繁项集 L_{k-1} 产生候选的 k 项集的集合 C_k，然后再通过扫描数据库确定对 C_k 中每一候选 k 项集的支持度，最后在该次扫描结束时得到频繁 k 项集 L_k，算法在 C_k 或 L_k 为空时终止。Apriori 产生的候选项集比 AIS 少得多，效率较高。Apriori 是关联分析的经典算法，很多算法都是其变种或改进。

DHP 算法利用散列表(Hash Table)产生候选项集，是对 Apriori 的直接改进。在遍历一次数据库得到候选 k 项集的支持度，得到频繁 k 项集后，将每一个事务的可能的$(k+1)$项集通过哈希规则形成散列表。散列表的每一栏包括所有通过散列规则映射到该栏中的项集的数目。根据结果的散列表，可以生成一个位向量，当散列表中对应该栏的值大于或者等于最小支持度时，对应的位置为 1，否则为 0。利用该向量可以过滤掉下一次生成候选时不必要的项集，即如果某候选项在向量中对应位置的值为 0，则舍弃，这对候选 2 项集的产生尤为有效，可以在第二次扫描时就大大减小候选项集的规模。在某些场合，DHP 的效率比

Apriori 明显提高。

PARTITION 算法主要针对大型数据库,分为两个步骤其一将目标数据库分为 n 个互不相交的子数据库 $D_1, D_2, \cdots, D_n$,每个 $D_i(i=1,2,\cdots,n)$都能存储在内存中。然后逐一把 D_i 读入内存并按一般算法发现频繁项集 L_i;再把所有的 L_i 合并为数据库的潜在频繁项集;其二计算潜在频繁项集在数据库中的支持度,得到频繁项集 L。

Sampling 算法的主要思想是对数据库 D 进行随机抽样得到抽样数据库 D',先以小于指定的最小支持度挖掘 D'的频繁项集 L',再在剩余数据集 $D-D'$中继续计算 L 中各元素的支持度,最后再以最小支持度求出 L。对于大部分情况可以得到所有的频繁项集,但有时会漏掉一些。可以对数据库进行二次扫描得到漏掉的频繁项集。多数情况下,此算法只需扫描数据库一次,最坏也只需扫描两次。

FP-growth 算法主要是采用一种新的数据结构 FP-tree,克服 Apriori 产生候选项集的缺点。该算法只扫描数据库两次,并且不用产生候选项集,提高了效率。

下面着重介绍 Apriori 和 FP-growth 算法。

8.2 Apriori

R. Agrawal 等在 1994 年提出了著名的 Apriori 算法,这是一种最有影响的挖掘布尔关联规则频繁项集的算法,得名于算法使用了频繁项集性质的先验知识。

Apriori 包含由候选项集(Candidate Itemset)产生频繁项集(Frequent Itemset),由频繁项集产生强关联规则(Strong Association Rule)两个步骤。

Apriori 使用逐层搜索的迭代方法,通过对数据库的多次扫描发现所有的频繁项集。在每一次扫描中只考虑具有同一长度 k(即项集中所含项的个数)的所有项集。算法的第一次扫描仅仅计算每个项具体值的数量,确定长度为 1 的频繁项集。在后继的每次扫描中,首先使用前一次扫描中找到的频繁项集 L_{k-1} 和 Apriori-gen 函数产生候选项集 q,接着扫描数据库,计算 C_k 中候选项集的支持度,最后确定哪些候选项集成为真正的频繁项集。重复上述过程直到再没有新的频繁项集出现。

Apriori 算法中,使用逐层迭代找出频繁项集的过程描述如下:

```
输入: 事务数据库 D,最小支持度阈值
输出: D 中的频繁项集 L
Begin
L1 = (large 1 - itemset);          /* 生成含有 1 项的频繁集 */
For (k = 2; Lk - 1≠Φ; k++) Do
 Begin
    Ck = apriori - gen(Lk - 1);
    For all transition t∈D Do
    Begin
      Ct = subset (Ck, t);         /* 产生事务 t 中包含的 k 项集 Ct */
      For all Candidate c∈Ct Do
     c.count++;                    /* 计数 */
    End;
    Lk = {c ∈Ck | c.count ≥min_sup}
```

```
13   End;
14 Answer = Lk;
15 End
```

产生候选项集的过程描述如下：

假定 L_{k-1} 中各项按某一次序排列，候选项集的产生由以下两个步骤组合而成。

连接步骤：Apriori-gen(Lk-1)的连接。

```
Begin
    insert into Ck
    select p.item1, p.item2, … ,p.itemk - 1, q.itemk - 1
    from Lk - 1,p, Lk - 1,q
    where p.item1 = q.item1, p.item2 = q.item2 , … ,p.itemk - 2 = q.itemk - 2, p.itemk - 1 < q.
    itemk - i1;
End;
```

剪枝步骤：Apriori-gen 的剪枝。

```
For all itemsets c∈Ck Do
    For all (k - 1) - subsets s of c Do
    if (s 不属于 Lk - 1) then
      delete c from Ck;
```

Apriori 首先产生频繁 1-项集 L_1，然后是频繁 2-项集 L_2，直到有某一 r 值使得 L_r 为空，则算法停止。其中第 k 次循环中，首先产生候选 k-项集的集合 C_k，C_k 中的每一个项集是对两个只有一个项不同的属于 L_{k-1} 的频繁集做一个(k-2)-连接产生的。L_1 中的项集是用来产生频繁项集的候选项集，最后的频繁项集 L_k 必须是 C_k 的一个子集。C_k 中的每个元素需在交易数据库中进行验证决定其是否加入 L_k，这里的验证过程是算法性能的一个瓶颈。这一方法要求多次扫描可能很大的交易数据库。Apriori-gen 的剪枝用来删除候选 L_k 中项集的子集不是频繁项集的集合。

对于具有一定规模的事务数据库而言，其蕴涵的候选项集数量特别巨大，同时一个事务内部也可能蕴涵许多的候选项集，这些成为制约 Apriori 算法性能的主要因素。

下面通过一个实例说明 Apriori 算法的实现过程。

(1) 设一个小型的交易数据库 D，如表 8.1 所示，设定最小支持度阈值为 2。

表 8.1　交易数据库 *D*

交易号	项集合	交易号	项集合
$T1$	$I1,I2,I5$	$T6$	$I2,I3$
$T2$	$I2,I4$	$T7$	$I1,I3$
$T3$	$I2,I3$	$T8$	$I1,I2,I3,I5$
$T4$	$I1,I2,I4$	$T9$	$I1,I2,I3$
$T5$	$I1,I3$		

(2) 扫描数据库 D，对每个候选项计数，生成 C_1，如表 8.2 所示。

(3) 比较候选项支持度计数与最小支持度，生成 L_1，如表 8.3 所示。

表 8.2 C_1

项集	支持度计数
{$I1$}	6
{$I2$}	7
{$I3$}	6
{$I4$}	2
{$I5$}	2

表 8.3 L_1

项集	支持度计数
{$I1$}	6
{$I2$}	7
{$I3$}	6
{$I4$}	2
{$I5$}	2

(4) 由 L_1 产生候选项集 C_2,如表 8.4 所示。
(5) 再次扫描数据库 D,对每个候选项计数,产生 L_2,如表 8.5 所示。
(6) 对 L_2 进行连接和剪枝,产生 C_3 即最终结果,如表 8.6 所示。

表 8.4 C_2

项集
{$I1$,$I2$}
{$I1$,$I3$}
{$I1$,$I4$}
{$I1$,$I5$}
{$I2$,$I3$}
{$I2$,$I4$}
{$I2$,$I5$}
{$I3$,$I4$}
{$I3$,$I5$}
{$I4$,$I5$}

表 8.5 L_2

项集	支持度计数
{$I1$,$I2$}	4
{$I1$,$I3$}	4
{$I1$,$I5$}	2
{$I2$,$I3$}	4
{$I2$,$I4$}	2
{$I2$,$I5$}	2

表 8.6 C_3

项集
{$I1$,$I2$,$I3$}
{$I1$,$I2$,$I5$}

其中,L_2 的连接与剪枝产生 C_3 的过程如下:

连接运算:

$$
\begin{aligned}
C_3 &= L_2 \infty L_2 \\
&= \{\{I1,I2\},\{I1,I3\},\{I1,I5\},\{I2,I3\},\{I2,I4\},\{I2,I5\}\} \infty \\
&\quad \{\{I1,I2\},\{I1,I3\},\{I1,I5\},\{I2,I3\},\{I2,I4\},\{I2,I5\}\} \\
&= \{\{I1,I2,I3\},\{I1,I2,I5\},\{I1,I3,I5\},\{I2,I3,I4\},\{I2,I3,I4\},\{I2,I3,I5\},\{I2,I4,I5\}\}
\end{aligned}
$$

剪枝运算:

{$I1$,$I2$,$I3$}的 2-项子集是{$I1$,$I2$}、{$I1$,$I3$}和{$I2$,$I3$}。

{$I1$,$I2$,$I3$}的所有 2-项子集都是 L_2 的元素。因此,保留{$I1$,$I2$,$I3$}在 C_3 中。

{$I2$,$I3$,$I5$}的 2-项子集是{$I2$,$I3$}、{$I2$,$I5$}和{$I3$,$I5$}。

{$I3$,$I5$}不是 L_2 的元素,因而不是频繁的。因此,删除 C_3 中的{$I2$,$I3$,$I5$}。

依此类推,剪枝后 C_3={{$I1$,$I2$,$I3$},{$I1$,$I2$,$I5$}}。

Apriori 的另一个实例如图 8.1 所示,其中最小支持度阈值为 2。

Apriori 利用候选项集和频繁项集的相互作用,得到了全部频繁项集,并通过对候选项集的剪枝,大大地减少了候选项集的规模,获得了令人满意的结果。然而,当面对挖掘对象

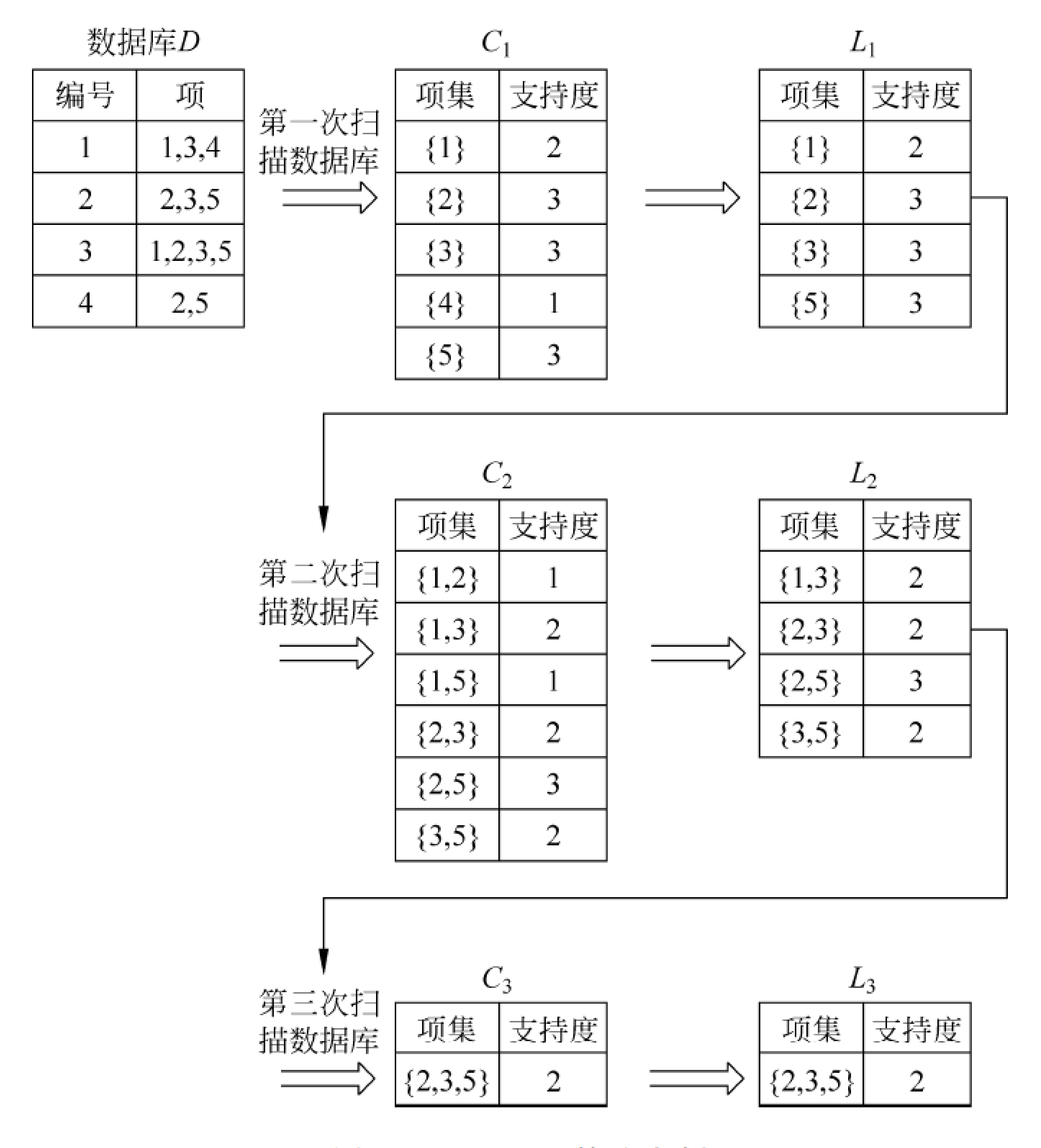


图 8.1　Apriori 算法实例

具有众多的频繁模式、长模式或者用户给定的最小支持度的阈值较低时，Apriori 仍然可能因为以下两个方面的巨大开销而面临困境。

(1)在处理大量的候选项集方面，如果算法得到了大量的频繁 1-项集 L_1，则在产生候选项集 C_2 时，会遇到 C_2 难以处理的情况。例如假设算法得到的频繁 1-项集 L_1 的数量是 10^4，则根据 Apriori 算法，产生的 2-项候选项集数量超过 10^7，由于候选项集 C_2 没有剪枝，所有候选项集都需要检验。此外，在面对频繁模式的规模较大时，同样会产生大量的候选项集需要检验。在内存等其他条件均为理想状态的情况下，这种由产生候选项集所产生的开销，无论采用什么技术都无法回避。所以，在有大量候选项集产生的情况下，Apriori 算法基本无法运行。

(2)采用的模式匹配方式，在检测大量的候选项集，特别是在挖掘长模式时，对数据库的重复扫描非常费时，大量时间消耗在内存与数据库的数据交换上。

由于上述原因，可以发现 Apriori 算法的瓶颈是候选项集的产生和测试过程。如果有一种算法能够对产生大量的候选项集进行有效的控制，将会极大地减少时间开销。

由于依赖从候选项集产生频繁项集的 Apriori 类算法具有先天的弱点，使得基于 Apriori 类算法的应用没有实质性突破。Jiawei Han 等提出了一种采用压缩的数据结构(FP-tree)存储关联规则挖掘所需的全部数据的新方法，通过对源数据的两次扫描，将数据存储到 FP-tree 结构，避免产生候选项集，极大地减少了数据交换和频繁匹配的开销，即所谓的无候选项集产生(Frequent Patterns Growth，FP-Growth)算法。

8.3 FP-Growth

FP-Growth 克服了必须产生候选项集的限制,提出关联规则挖掘的新思路,主要改进体现在:

- 构造一种新颖的、紧凑的数据结构 FP-tree。它是一种扩展的前缀树结构,存储关于频繁模式数量的重要信息。树中只包含长度为 1 的频繁项作为叶结点,并且那些频度高的结点更靠近树的根结点,因此频度高的项比那些频度低的项有更多的机会共享同一个结点。
- 开发基于 FP-tree 的模式片断成长算法,从长度为 1 的频繁模式开始,只检查它的条件模式及构建它的条件模式树,并且在这个树上递归地执行挖掘。模式的成长通过联合条件模式树新产生的后缀模式实现。由于事务处理中的频繁项都对应着频繁树中的路径进行编码,模式的成长确保了结果的完整性。因此,FP-Growth 不像 Apriori 那样需要产生再测试。挖掘的主要操作是计算累加值和调整前缀树,这种开销通常远远小于 Apriori 类算法中候选项集的产生和模式匹配操作。
- 采用基于分区的搜索,通过分割,而不是 Apriori 类算法的自下向上地产生频繁模式的集合。将发现长频繁模式的问题转化为寻找短模式,然后再用后缀连接的方法,避免产生长的候选项集。

FP-tree 的结构包括一个标识成 null 的根、一个由频繁项组成的头表和一组项的前缀子树组成根的子孙。树中的每个结点包括项名(Itemname)、计数(Count)和结点链接(Node Link)三个域。其中,项名标识结点所代表的项;计数标识树中到达这个结点的路径所代表的事务处理的数目;结点链接指向树中下一个同名结点,如果没有同名结点则指向空。头表的每条记录包含两个域,即项名和结点链接的头。结点链接的头指向树中第一个同名的结点。

FP-tree 只保存满足最小支持度的项的集合。所以,首先需要知道哪些项符合条件,即构造头表。对数据库进行第一次扫描得到满足最小支持度的项并按降序排列在头表中。在得到头表之后,对数据库进行第二次扫描,对每个事务处理包含的频繁项按照其在头表的先后顺序插入到树。插入到树的事务处理的频繁项自然是树的一个路径,但如果树中存在其他与新路径完全相同或部分相同的路径,则需要将两个路径全部或部分合并,将事务处理插入到 FP-tree 中的函数 insert-tree([p/P],T)是算法中一个非常关键的部分。

FP-tree 是一个高度压缩的结构,存储用于频繁模式挖掘的全部信息,由于一个以 a_1 作为前缀的单一路径 $a_1 \rightarrow a_2 \rightarrow \cdots \rightarrow a_k$ 代表所有那些最大的频繁集形式为 $a_1 \rightarrow a_2 \rightarrow \cdots \rightarrow a_k (1 \leqslant k \leqslant n)$ 的事务处理,所以 FP-tree 远远小于源数据和在关联规则挖掘过程中产生的候选项集的大小。同时,频繁项集中的项以支持度降序排列,支持度越高的项与 FP-tree 树根越接近,因此有更多的项是共享的。

FP-Growth 算法描述如下:

```
输入: 事务数据库 D,最小支持度阈值 minsup
输出: 全部频繁项的集合
1. 构造 FP - tree
扫描数据库一次.得到频繁项的集合 F 和其支持度,按照支持度对 F 降序排序,产生频繁项表 L.
创建 FP - tree 的根结点,标记为 Null.对于数据库的每一个事务,选择频繁项,并按 L 中的次序排序.
```

设排序后的频繁项表为[p/P],其中 p 是第一个元素,而 T 是剩余元素的表.调用 insert－tree([p/P],T).该过程执行情况如下:如果 T 有子女 N 使得 N.item－name＝p. item－name,则 N 的计数加 1;否则创建一个新结点 N,将其计数设置为 1,链接到其父结点 T,并且通过结点链结构将其链接到具有相同项名的结点;如果 P 非空,递归调用 insert－tree(P,N).

2. 在 FP 树中挖掘频繁模式

输入:事务数据库 D 的 FP 树和最小支持度阈值 minsup

输出:所有的频繁模式的集合

方法:调用 FP－Growth (FP－Tree,null)

```
Procedure FP－Growth (Tree,α)
if (Tree 只包含单路径 P) then
    对路径 P 中结点的每个组合(记为 β)
    生成模式 β∪α,支持度＝β 中所有结点的最小支持度
else 对 Tree 头上的每个 ai do
    生成模式 β＝ ai ∪α,支持度＝ai.support;
    构造 β 的条件模式库和 β 的条件 FP 树 Treeβ;
    if Treeβ≠ϕ
    then call FP－Growth (Treeβ,β)
```

这是一个递归调用函数,根据 FP-tree 的属性,对于给定的数据源和支持度阈值,算法可以得到所有满足条件的频繁项集。

假设最小支持度阈值为 0.5,交易数据库如表 8.7 所示。

表 8.7　交易数据库

交 易 编 号	购　物　项	排序后的频繁项
100	f,a,c,d,g,i,m,p	f,c,a,m,p
200	a,b,c,f,l,m,o	f,c,a,b,m
300	b,f,h,j,o	f,b
400	b,c,k,s,p	c,b,p
500	a,f,c,e,l,p,m,n	f,c,a,m,p

第一次扫描数据库,得到频繁 1 项集,然后按照频度降序排列。再次扫描数据库,生成的 FP-tree 如图 8.2 所示。

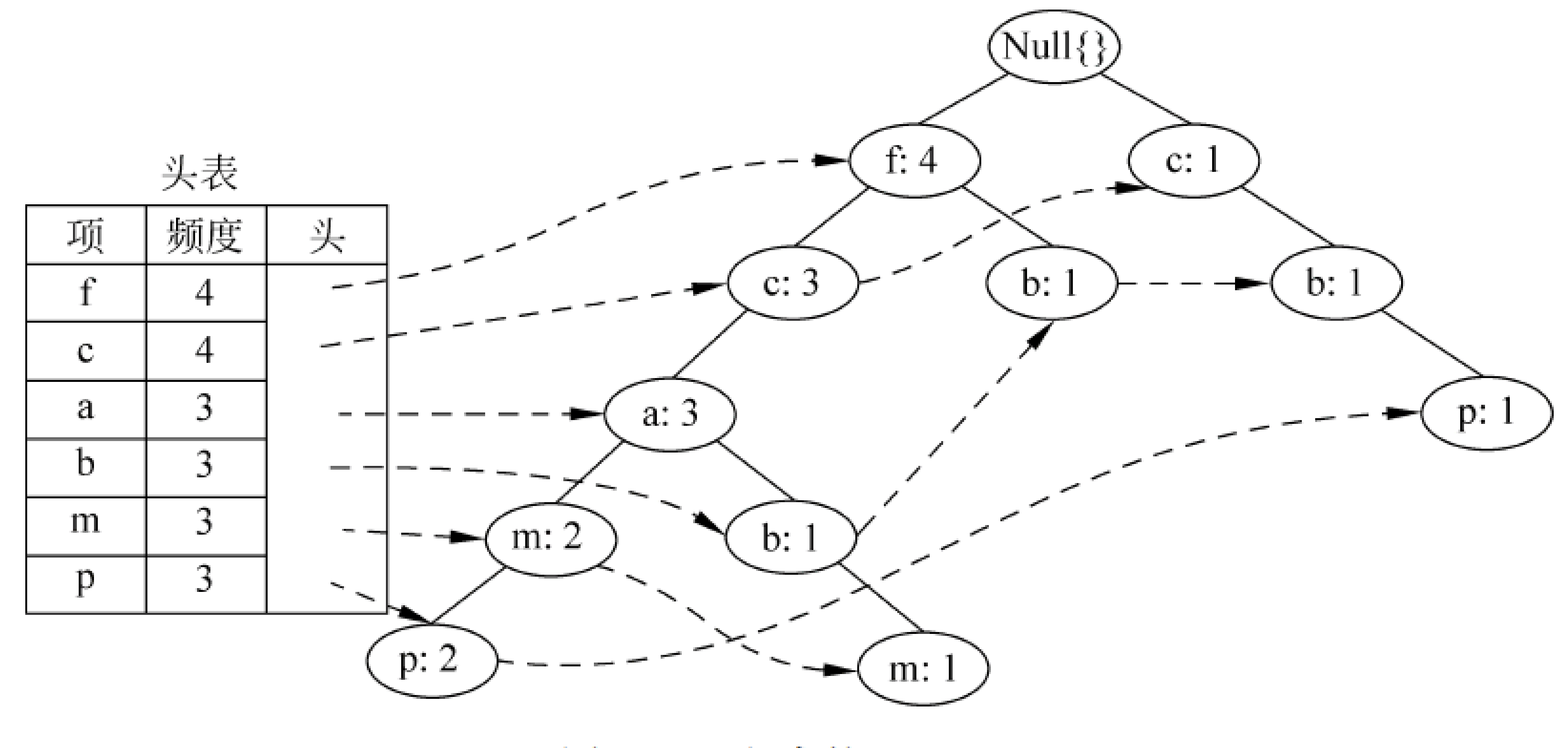


图 8.2　生成的 FP-tree

由图 8.2 可知,对任意频繁项 a_i,顺着 a_i 的结点链,从 a_i 的头开始,可以找到包含 a_i 的所有频繁模式。

第9章 Web 挖 掘

随着 Internet 的巨大发展，数以亿计的静态网页和动态网页汇聚了人类无穷的智慧，海量 Web 数据包含了无数的链接、访问路径和内容。据统计，目前全世界将近 90%的信息是 Web，而且其数量和重要性仍在与日俱增。Web 具有异构、半结构化和动态等特点。

1. 异构

从数据库的角度，Web 网站上的信息也可以看作一个数据库，一个更大、更复杂的数据库。每一个 Web 站点就是一个数据源，每个数据源都是异构的，因而每一站点之间的信息和组织都不一样，这就构成了一个巨大的异构数据库环境。如果想要利用这些 Web 进行挖掘，首先，必须要研究站点之间异构 Web 的集成问题，只有将它们都集成起来，提供给用户一个统一视图，才有可能从巨大的数据源中获取所需的东西。其次，还要解决 Web 查询问题，因为如果所需的 Web 无法获取，对它们的分析、集成和处理就无从谈起。

2. 半结构化

Web 与传统的数据库中的数据不同，传统的数据库都有一定的数据模型，可以根据模型具体描述特定的数据。而 Web 非常复杂，没有特定的模型描述，每一站点的 Web 都各自独立，并且本身具有自述性和动态可变性。因而，Web 虽具有一定的结构性，但因自述性的存在，故是一种非完全结构化的数据，称为半结构化数据。半结构化是 Web 的最大特点。

3. 动态性

Web 不仅每天都以极快的速度增长，而且也在不断地动态变化。因此需要借助数据仓库技术，以保存动态更新的 Web。

综上所述，从海量 Web 中真正发现知识存在一定的困难，主要体现在：

(1) Web 数量太庞大，例如 Web Informall 达 15TB，而且仍在迅速增加。

(2) Web 复杂性高于任何传统的文本，例如 Web 分类需要预处理，Web 缺乏统一的结构。Web 可以被看作一个巨大的数字图书馆，然而这一图书馆中的大量信息并没有按照任何排序进行组织，没有分类索引，更没有标题、作者扉页和目次等索引，在其中搜索所需信息极具挑战性。

(3) Web 面向广泛的用户群，且仍在不断扩展。不同用户有着不同的背景、兴趣和使用目的。大部分用户并不了解 Web 的结构，不清楚搜索的高昂代价，极易在“黑暗”中迷失方向，在“跳跃式”访问中烦乱不已和在等待中失去耐心。

(4) Web 只有很少的一部分是相关或有用的。据说 99%的 Web 相对于 99%的用户是无用的，虽然这看起来不是很明显，但每个人只关心很小一部分 Web 确是事实。

由于 Web 的复杂性和动态性，难以搜索、发现和利用 Web 中蕴藏的大量知识，因此 Web 挖掘是知识发现领域的重要课题之一。

9.1 概述

9.1.1 定义和分类

1. 定义

Web 挖掘由数据挖掘发展而来，但 Web 挖掘比传统的数据挖掘复杂，涉及数据挖掘、计算机语言学和信息科学等多领域。研究者从不同的角度出发，对 Web 挖掘的定义有所不同。

各种文献中，常见的 Web 挖掘定义包括如下几个方面。

1）描述性的定义

Web 挖掘是指使用数据挖掘技术在 WWW 数据中发现潜在的、有用的模式或信息。Web 挖掘是一项综合技术，覆盖了多个研究领域，包括 Web 技术、数据库、数据挖掘、计算语言学、信息获取、统计学以及人工智能等。

2）抽象化的定义

一般地，Web 挖掘是指从大量 Web 集合 C 中发现隐含的模式 p。如果将 C 看作输入，将 p 看作输出，则 Web 挖掘就是一个从输入到输出的映射，即 $\xi: C \rightarrow p$。

3）概括性的定义

Web 挖掘是从与 WWW 相关的资源和行为中抽取感兴趣的、潜在的、有用的模式和隐含信息。

Web 挖掘可在很多方面发挥作用，如搜索引擎结构挖掘、确定权威页面、Web 文档分类、Web 日志挖掘和智能检索等。

2. 分类

根据兴趣的不同，一般的 Web 挖掘分为 Web 内容挖掘（Web Content Mining）、Web 结构挖掘（Web Structure Mining）和 Web 使用挖掘（Web Usage Mining）三类，如图 9.1 所示。

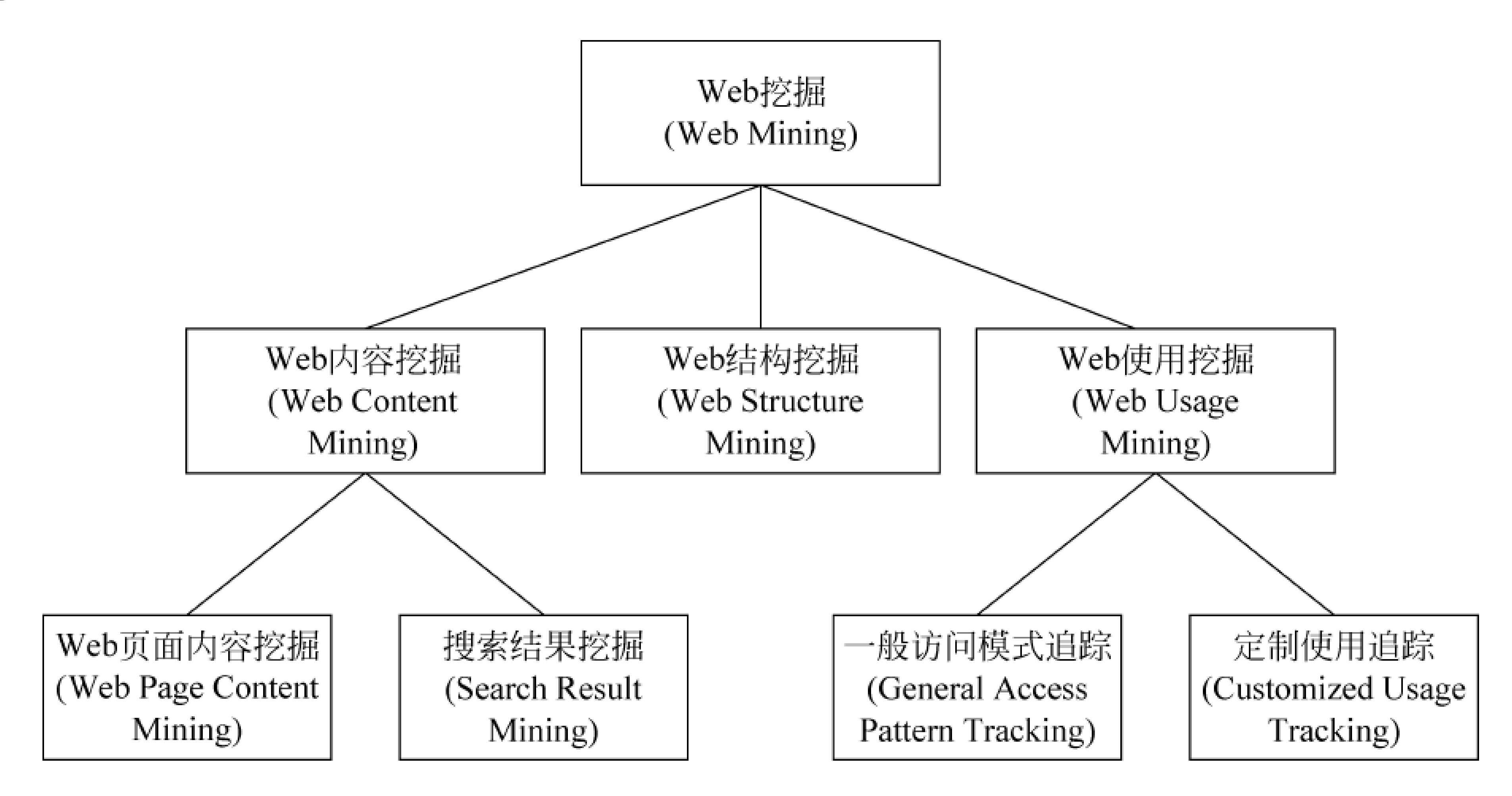


图 9.1　Web 挖掘的分类

1）Web 内容挖掘

Web 内容挖掘是指从 Web 内容中发现有用信息，Web 信息五花八门，如政府信息服务、数字图书馆、电子商务数据以及其他各种通过 Web 访问的数据库。Web 内容挖掘对象包括文本、图像、音频、视频、多媒体和其他各种类型的数据，其中针对非结构化文本的 Web 挖掘被归类到基于文本的知识发现，也称文本数据挖掘或文本挖掘。现在，Web 多媒体数据挖掘成为一个热点。

一般地，Web 内容挖掘可以从两种不同的角度展开。从资源查找（Information Retrieval，IR）的角度看，Web 内容挖掘的任务是从用户的角度出发，提高信息质量和帮助用户过滤信息；从数据库的角度看，Web 内容挖掘的任务主要是对 Web 数据进行集成、建模，以支持复杂的 Web 查询。

（1）从资源查找的角度挖掘非结构化文档

非结构化文档主要指 Web 上的自由文本，包括小说、新闻等。在这方面的研究相对比较多，大部分研究都是建立在词汇袋（Bag of Words）或向量表示法（Vector Representation）的基础上，这些方法将单个词汇看成文档集合中的属性，只从统计的角度将词汇孤立地看待而忽略该词汇出现的位置和上下文环境。属性可以是布尔型，即词汇是否在文档中出现；也可以是频度，即词汇在文档中的出现频率。词汇袋方法的一个弊端是自由文本中的词汇量非常大，难以处理。为了解决上述问题，可采用信息增益、交叉熵和差异比等方法减少属性。另外，一种比较有意义的方法是潜在语义索引（Latent Semantic Indexing），通过分析不同文档中相同主题的共享词汇，找到其共同的词根，利用公共词根代替所有词汇，以此减少维空间，例如 informing、information、informer 和 informed 可以用它们的词根“inform”表示，这样可以减少属性集合的规模。

其他的属性表示法还包括词汇在文档中的出现位置、层次关系、使用短语，使用术语和命名实体等。目前还没有研究表明一种表示法明显优于另一种。

（2）从资源查找的角度挖掘半结构化文档

与非结构化文档相比，Web 上的半结构化文档挖掘是指对加入了 HTML、超链接等附加结构的信息进行挖掘，其应用包括超链接文本的分类、聚类、发现文档之间的关系以及提取半结构化文档中的模式和规则等。

（3）从数据库的角度挖掘非结构化文档

数据库技术应用于 Web 挖掘主要是为了解决 Web 信息的管理和查询。这些问题可以分为三类：Web 信息的建模和查询；信息抽取与集成；Web 站点构建和重构。

从数据库的观点进行 Web 内容挖掘主要是建立 Web 站点的数据模型并集成，以支持复杂查询，而不只是简单的基于关键词的搜索。这要通过找到 Web 文档的模式、建立 Web 数据仓库或 Web 知识库或虚拟数据库实现。

主要利用 OEM（Object Exchange Model，对象交换模型）将半结构化数据表示成标识图。OEM 的每个对象都有对象标识（OID）和值，值可以是原子类型，如整型、字符串、gif 和 html 等，也可以是一个复合类型，以对象引用集合的形式表示。由于 Web 数量非常庞大，从应用的角度考虑，很多研究只处理半结构化数据的一个常用子集。一些有意义的应用是建立多层数据库，每一层是其下面层次的概化，这样则可以进行一些特殊的查询和信息处理。由于数据的表示方法较特殊，其中包含了关系层次和图形化的数据，所以大部分建立在

扁平数据集合之上的数据挖掘方法不能直接使用,目前已经开展针对多层数据库挖掘算法的研究。

2) Web 结构挖掘

Web 结构挖掘的对象是 Web 本身的超链接,即对 Web 文档结构的挖掘。对于给定的 Web 文档集合,应该能够通过算法发现它们之间连接情况的有用信息,文档之间的超链接反映包含、引用或者从属关系,引用文档对被引用文档的说明往往更客观、更概括和更准确。

Web 结构挖掘在一定程度上得益于社会网络和引用分析的研究。把 Web 之间的关系分为 incoming 和 outgoing 连接,运用引用分析方法找到同一网站内部以及不同网站之间的链接关系。在 Web 结构挖掘领域最著名的算法是 HITS 和 PageRank 算法,其共同点是通过计算 Web 之间超链接的质量,从而得到页面权重。著名的 Google 搜索引擎采用了该类算法。此外,Web 结构挖掘的另一个尝试是在 Web 数据仓库环境下,通过检查同一台服务器上的本地连接衡量 Web 结构,挖掘 Web 站点的完全性,在不同的 Web 数据仓库中检查副本以帮助定位镜像站点,通过发现针对某一特定领域的超链接的层次属性探索信息流动如何影响 Web 站点的设计。

3) Web 使用挖掘

即 Web 使用记录挖掘,在电子商务领域具有重要意义。它通过挖掘相关的 Web 日志,发现用户访问 Web 页面的模式,通过分析日志,可以识别用户的忠诚度、喜好和满意度,可以发现潜在用户,增强站点的服务竞争力。除了服务器的日志外,还包括代理服务器日志、浏览器端日志、注册信息、用户会话信息、交易信息、Cookie 中的信息、用户查询和鼠标点击流等一切用户与站点之间可能的交互记录。可见 Web 使用记录的数据量非常巨大,而且数据类型也相当丰富。

根据对数据源的处理方法不同,Web 使用挖掘可以分为两类:其一是将 Web 使用记录转换并导入传统的关系表,再使用数据挖掘算法对关系表数据进行常规挖掘;其二是将 Web 使用记录直接预处理再进行挖掘。Web 使用挖掘的一个有趣问题是在多个用户使用同一台代理服务器环境下如何标识某个用户,如何识别属于该用户的会话和使用记录,这个问题看起来不大,但却在很大程度上影响着挖掘质量。通常经典的数据挖掘算法都可以直接用于 Web 使用挖掘。但是,为了提高挖掘质量,可采用改进算法,如复合关联规则算法、改进的序列发现算法等。

Web 使用挖掘的应用主要体现在:

- 个性挖掘

针对单个用户的使用记录对该用户进行建模,结合用户基本信息分析其使用习惯、个人喜好,目的是在电子商务环境下为该用户提供与众不同的个性化服务。

- 系统改进

Web 服务(数据库、网络等)的性能和其他服务质量是衡量用户满意度的关键指标,Web 使用挖掘可以通过用户的拥塞记录发现站点的性能瓶颈,提示站点管理者改进 Web 缓存策略、网络传输策略、流量负载均衡机制和数据分布策略等。此外,还可以通过分析网络的非法入侵数据发现系统弱点,提高网站安全性,这在电子商务环境下尤为重要。

- 站点修改

站点的结构和内容是吸引用户的关键。Web 使用挖掘通过挖掘用户的行为记录和反

馈信息为站点设计者提供改进依据,例如页面链接如何组织、哪些页面能够直接访问等。

• 商业智能

用户怎样使用Web站点的信息无疑是电子商务销售商关心的重点,用户一次访问的周期可分为被吸引、驻留、购买和离开四个阶段。Web使用挖掘可以通过分析用户点击流等Web日志挖掘用户行为的动机,帮助销售商制定合理的销售策略。

• Web特征描述

通过用户对站点的访问情况统计各个用户的Web交互情况,描述用户的访问特征。

目前,Web挖掘存在的不足主要体现在:第一,Web内容挖掘、Web结构挖掘和Web使用挖掘相互独立,没有互相利用有机结合。Web内容挖掘算法没有充分利用Web文档中的结构信息和超链接信息;第二,Web内容挖掘几乎没有利用语义信息。目前的Web内容挖掘大都是基于信息提取,而没有在获取Web文档的语义信息基础上建立挖掘的理论和算法,这使得挖掘结果的精度较差,质量不够理想;第三,Web挖掘算法尚不能有效处理海量数据。这也是传统数据挖掘面临的难题之一。

基于Web的数据挖掘和传统的基于数据仓库的数据挖掘有着不同的含义。根据W. J. Frawle和G. P. Shapir等人的定义,一般的数据挖掘指从大型数据仓库的数据中提取人们感兴趣的知识,而这些知识是隐含的,事先未知的、潜在的有用信息,它侧重于从已有的信息中提取规律性的知识。而Web挖掘的研究对象是以半结构化和非结构化的文档为中心的Web,这些数据没有统一的模式,数据的内容和表示互相交织,内容基本上没有语义信息,仅仅依靠HTML语法对数据进行结构上的描述。为了对这种半结构化数据进行分析和处理,Web挖掘必须和其他研究手段结合起来。由于涉及到很多的知识领域,Web挖掘现在是多个研究方向的交汇点,包括信息获取、人工智能、机器学习、模式识别、统计学和自然语言处理等。

9.1.2 主要技术

随着社会的日益信息化,人们越来越强烈地希望运用自然语言同计算机交流,建立一种人与机器之间密切而友好的关系,使之能进行高级的信息传递与认知活动。自然语言理解是计算机科学中的一个引人入胜和富有挑战性的课题。从计算机科学特别是从人工智能的观点看,自然语言理解的任务是建立一种计算机模型,它能够给出像人那样理解、分析并回答自然语言(即人们日常使用的各种通俗语言)的结果。自然语言处理就是研究如何能让计算机理解并生成人们日常所使用的语言(如汉语、英语等),让计算机懂得自然语言的含义,并对人给计算机提出的问题,通过对话的方式,用自然语言进行回答。自然语言理解系统可以用作专家系统、知识工程、情报检索、办公自动化的自然语言人机接口,具有重要的实用价值。

1. 发展与演变

由于对自然语言理解的需求,因此对自然语言处理的研究在电子计算机问世之初就开始了。20世纪40年代末期曾有学者提出用计算机进行自然语言翻译的构想,并于20世纪50年代初开展了机器翻译试验。第一代翻译系统以词汇转换为主,很少进行句法分析,还不能称作“理解”。

到了20世纪60年代，乔姆斯基的转换生成语法得到广泛的认可，对句子的分析是利用短语结构规则自顶向下或自底向上地生成句法树，从而得到句子的句法结构。转换生成文法把机器翻译带入句法分析的时代，也使得对自然语言的处理上升到新的水平。由于认识到生成语法缺少表示语义知识的手段，不利于自然语言理解，在20世纪70年代随着认知科学的兴盛，学者们纷纷从语义的角度出发，提出语义理论，在自然语言处理中大量引进语义、语境以及语用的分析技术。20世纪60年代末期，M. R. Qullian提出了语义网络理论，用于描述概念之间的关系；C. J. Fillmore提出了格语法，用语义格和深层格框描述句义；1973年，Roger Schank提出了概念从属理论（Conceptual Dependency，CD理论），描述句义和语义；1975年，Marvin Minsky提出了框架理论，用于描述事物或概念状态。这些理论经过各自的发展，逐渐开始趋于相互融合。

到20世纪80年代一批新的语法理论脱颖而出，具有代表性的包括词汇功能语法（Lexical Function Grammar，LFG）、功能合一语法（Functional Unification Grammar，FUG）、广义短语结构语法（Generalized Phrase Structure Grammar，GPSG）等。虽然，这些基于规则的分析方法基本上解决了单个句子的分析，但是还很难覆盖全面的语言现象，特别是对于整个段落或篇章的理解还无从下手。

20世纪90年代，在自然语言处理领域中，出现了基于语料库的方法，对大规模真实文本进行处理。这些方法包括统计和基于实例的方法等。通过词法、句法、语义等多层次的加工从未经处理的生语料中获取各种语言知识、情景知识和语境知识等，然后利用这些知识对语言进行分析理解。因此基于知识的方法成为主流。从整个自然语言理解的发展历程来看，自然语言理解经历了从单纯依靠语法规则到语义分析与句法分析相结合，最后利用知识来消除歧义的过程。

语料库是大量文本的集合，计算机出现后，语料可以被方便地存储起来，利用计算机查找也很容易。随着电子出版物的出现，语料采集也不再困难。最早于20世纪60年代编制的Brown和LOB两个计算机语料库，分别具有100万词次的规模。进入20世纪90年代语料库达几十个之多，像DCI、ECI、ICAME、BNC、LDC和CLR等，其规模最高达到10^9数量级。

对语料库的研究分为3个方面：工具软件的开发、语料库的标注和基于语料库的语言分析方法。采集后未经处理的生语料不能直接提供有关语言的各种知识，只有通过词法、句法和语义等多层次的加工才能使知识获取成为可能。加工方法就是在语料中标注各种记号，标注的内容包括每个词的词性、语义项、短语结构、句型和句间关系等。随着标注程度的加深语料库逐渐熟化，成为一个分布的、统计意义上的知识源。利用这个知识源可以进行许多语言分析工作，例如根据从已标注语料中总结出的频度规律可以给新文本逐词标注词性、划分句子成分等。

语料库提供的知识是用统计方法表示的，而不是确定性的，随着规模的扩大，旨在覆盖全面的语言现象。但是对于语言中基本的确定性的规则仍然用统计强度的大小去判断，这与人们的常识相违背。这种“经验主义”的不足要靠理性的方法来弥补。两类方法的融合也正是当前自然语言处理的发展趋势。

自然语言理解的发展可以分为第一代系统和第二代系统两个阶段。第一代系统建立在对词类和词序分析的基础之上，分析中经常使用统计方法；第二代系统则开始引入语义甚至语用和语境的因素，几乎完全抛开了统计方法。

第一代自然语言理解系统又可分为四种类型。

1) 特殊格式系统

早期的自然语言理解系统大多数是特殊格式系统,根据人机对话内容的特点,采用特殊的格式来进行人机对话。1963 年,林德赛(R. Lindsay)在美国卡内基学院用 IPL-V 表处理语言设计的 SAD-SAM 系统,就采用了特殊格式进行关于亲属关系方面的人机对话,系统内建立了一个关于亲属关系的数据库,可接收关于亲属关系方面的问题的英语句子提问,并用英语作出回答;1968 年,波布洛(D. Bobrow)在美国麻省理工学院设计了 STUDENT 系统,这个系统把高中代数应用题中的英语句子归纳为一些基本模式,由计算机来理解这些应用题中的英语句子,列出方程求解,并给出答案。20 世纪 60 年代初期,格林(B. Green)在美国林肯实验室建立了 BASEBALL 系统,也使用 IPL-V 表处理语言,系统的数据库中存储了关于美国 1959 年联邦棒球赛得分记录的数据,可回答有关棒球赛的一些问题。该系统的句法分析能力较差,输入句子十分简单,没有连接词,也没有比较级形式的形容词和副词,主要靠一部机器词典进行单词的识别,使用了 14 个词类范畴,所有的问题都采用一种特殊的规范表达式回答。

2) 以文本为基础的系统

特殊格式系统中格式的限制,带来了诸多不便。因为就一个专门领域而言,最方便的还是使用不受特殊格式限制的系统进行人机对话,因此后来出现了以文本为基础的系统,1966 年西蒙(R. F. Simmons)、布尔格(J. F. Burger)和龙格(R. E. Long)设计的 PROTOSYNTHEX-I 系统,就是以文本信息的存储和检索方式工作的。

3) 有限逻辑系统

有限逻辑系统进一步改进了以文本为基础的系统。在这种系统中,自然语言的句子以某种更加形式化的记号来替代,这些记号自成一个有限逻辑系统,可以进行某些推理。1968 年,拉菲尔(B. Raphael)在美国麻省理工学院用 LISP 语言建立了 SIR 系统,针对英语提出了 24 个匹配模式,把输入的英语句子与这些模式相匹配,从而识别输入句子的结构,在从存储知识的数据库到回答问题的过程中,可以处理人们对话中常用的一些概念,如集合的包含关系、空间关系等等,并可进行简单的逻辑推理,并且机器能在对话中进行学习,记住已学过的知识,从事一些初步的智能活动。1965 年,斯莱格勒(J. R. Slagle)建立了 DEDUCOM 系统,可在情报检索中进行演绎推理。1966 年,托普逊(F. B. Thompson)建立了 DEACON 系统,通过英语管理一个虚构的军用数据库,设计中使用了环结构和近似英语的概念进行推理。1968 年,凯罗格(C. Kellog)在 IBM360/67 计算机上,建立了 CONVERSE 系统,该系统能根据关于美国 120 个城市的 1000 个实际的文件来进行推理。

4) 一般演绎系统

一般演绎系统使用某些标准数学符号(如谓词演算符号)表达信息。逻辑学家们在定理证明工作上取得的全部成就,就可以用来作为建立有效的演绎系统的依据,从而能够把任何一个问题用定理证明的方式表达出来,并实际地演绎出所需要的信息,用自然语言做出回答。一般演绎系统可以表达那些在有限逻辑系统中不容易表达出来的复杂信息,进一步提高了自然语言理解系统的能力。1968 至 1969 年,格林和拉菲尔建立的 QA2 和 QA3 系统,采用谓词演算的方式和格式化的数据进行演绎推理,解答问题,并用英语作出回答,这是一般演绎系统的典型代表。

1970 年以来，出现了第二代自然语言理解系统，这些系统绝大多数是程序演绎系统，大量地进行语义、语境和语用的分析。其中比较有名的是 LUNAR、SHRDLU、MARGIE、SAM、PAM 等系统。

LUNAR 系统是伍兹(W. Woods)于 1972 年设计的一个自然语言情报检索系统。这个系统采用形式提问语言(formal query language)表示所提问的语义，从而对提问的句子作出语义解释，最后把形式提问语言用于数据库，产生问题的回答。

SHRDLU 系统是维诺格拉德(T. Winograd)于 1972 年在美国麻省理工学院建立的一个用自然语言指挥机器人动作的系统。该系统把句法分析、语义分析和逻辑推理结合起来，大大地增强了系统在语言分析方面的功能。该系统对话的对象是一个具有简单的“手”和“眼”的玩具机器人，它可以操作放在桌子上的不同颜色、尺寸和形状的玩具积木，如立方体、棱锥体和盒子等，机器人能够根据操作人员的命令把这些积木捡起来，移动它们去搭成新的积木结构，在人机对话过程中，操作人员能获得发送给机器人的各种视觉反馈，实时地观察机器人理解语言和执行命令的情况。在电视屏幕上还可以显示出这个机器人的模拟形象以及它同一个真人在电传机上自由地用英语对话的生动情景。

MARGIE 系统是杉克(R. Schank)于 1975 年在美国斯坦福大学人工智能实验室研制的。该系统的目的在于提供一个自然语言理解的直观模型。系统首先把英语句子转换为概念依存表达式，然后根据系统中有关信息进行推理，从概念依存表达式中推演出大量的事实。由于人们在理解句子时，总要牵涉到比句子的外部表达多得多的内容，因此，该系统的推理有 16 种类型，如原因、效应、说明和功能等。最后，把推理的结果转换成英语输出。

SAM 系统是阿贝尔森(R. Abelson)于 1975 年在美国耶鲁大学建立的。这个系统采用脚本(Script)理解自然语言编写的故事。所谓脚本，就是用来描述人们活动(如上饭馆、看病)的一种标准化的事件系列。

PAM 系统是威林斯基(R. Wilensky)于 1978 年在美国耶鲁大学建立的另一个理解故事的系统。PAM 系统能解释故事情节，回答问题，进行推论，做出摘要。它除了“脚本”中的事件序列之外，还提出了计划(Plan)作为理解故事的基础。所谓计划就是故事中的人物为实现其目的所要采取的手段。如果要通过“计划”来理解故事，就要找出人物的目的以及为完成这个目的所采取的行动。系统中设有一个“计划库”(Plan Box)，其中存储着有关各种目的的信息以及各种手段的信息。这样，在理解故事时，只要找到故事中有关情节与计划库中存储的信息相匹配的部分，就可以理解到这个故事的目的是什么。当把一个一个的故事情节与脚本匹配出现障碍时，由于“计划库”中可提供关于一般目的的信息，就不致造成故事理解的失败。例如，营救一个被暴徒抢走的人，在“营救”这个总目的下列出若干个子目的，包括到达暴徒的巢穴以及杀死暴徒的各种方法，就可以预期下一步的行为。同时能根据主题推论目的。例如，输入故事：“约翰爱玛丽；玛丽被暴徒抢走了。”PAM 系统即可预期约翰要采取行动营救玛丽。故事中虽然没有这样的内容，但是，根据计划库中的“爱情主题”，可以推出“约翰要采取行动营救玛丽”的情节。

上述系统都是书面的自然语言理解系统，输入和输出都是用书面文字。口头的自然语言理解系统，还牵涉到语音识别和语音合成等复杂技术，显然是更加困难的课题，口头自然语言理解系统的研究近年来亦有进展。

2. 现代汉语的研究现状

汉语研究基本上都是从概率统计向着汉语的语义研究方向发展。语义是汉语理解的一条蹊径,以词义为基础,与句法、规则相结合,以句子为突破单位。国内对汉语的信息处理研究项目主要有以下三个流派。

第一是以传统计算语言学为基本理论,从词素分析入手,进而研究词-短语(词组)-语段-句子。概括地说,传统计算语言学的种种理论和方法,都以语料统计为基础,因此还需要结合语言规则,例如借鉴了西方计算语言学的众多理论和方法如短语结构语法、扩充转移网络、从属关系语法和配价语法等。

第二是黄曾阳提出的概念层次网络理论(Hierarchical Network of Concepts,HNC 理论)。HNC 理论认为,自然语言理解的关键是描述人的语言感知过程的适当模式,试图建立一种模拟大脑语言认知过程的自然语言的计算机理解处理模式。HNC 把自然语言要表述的知识划分为概念、语言和常识三个独立的层面,并为此建立不同的知识库,通过建立局部和全局两类联想脉络来帮助计算机理解自然语言。

第三是基于内涵模型的语义分析,该理论是由陆汝占教授提出的。它将汉语表达式抽象成数学表达式,恰当地表示内涵和外延,然后把这些语义表示在计算机内进行处理,即把汉语表达式与计算机数据结构之间直线联结,改变为汉语表达式-抽象数学表示-数据结构三者的间接联结。具体设想是:先构造一种句子的逻辑式之间的中介形式“函子”(Functor),以表示谓语动词连同支配成分一起构成的语句核心,表示句义的基本要素。函子加上时态、模态算子就可以表示语态,构成句子的基本逻辑含义。

以上对汉语的研究面临着不同的问题。第一种流派较好地处理了汉语的表层语法结构,但是面临着如何集成和如何解决词义、句子的问题;第三种流派,理论设计还较粗略,趋向于把自然语言的表示数学化,虽然用这一理论已经解决了一些实用问题,但是对自然语言本身的理解做得不够,要证明它可以适用于整个现代汉语,还需要进一步推敲、实验和细化。

对汉语的处理不仅依赖于语言的表层结构表述,更重要的是语言深层结构的表述。第二种流派 HNC 提出了深层次的语言结构的表示,语言抽象表示的概念化和层次化都适合汉语的研究,但其实现比较庞大复杂,需要建立庞大的知识库,从总体上检验和完善其理论和技术设计的可行性,因此该理论需要长期的实践验证。此外,鲁川提出了汉语的意合网络,给出了语义的组合知识,但是对于知识的获取涉及甚少。董振东提出的知网(How-Net)建立了一个以揭示概念与概念之间以及概念所具有的属性之间的关系为基本内容的常识知识库,但是对概念的描述停留在词汇层面上,没有足够的知识适合于计算机推理。俞士汉等初步建成了“现代汉语语法电子词典”,提出了现代汉语词语分类体系,但只是服务于语言信息处理。

3. 相关理论

1) 国外的研究现状

国外关于自然语言理解方面的研究起步较早,一些卓有成就的语言学家、逻辑学家和心理学家都在自然语言理解的语法、句法及语义分析方面提出了一些较为系统的理论和方法。下面介绍一些比较有影响的理论。

(1) 形式语言

1957 年美国 Chomsky 提出了形式语言理论。在该理论中,语言被看成是一个抽象的

符号系统，Chomsky 把它定义为：按一定的规律构成的句子或符号串的有限的或无限的集合，记为 L。一种语言的文法 G 是一种格式，用来说明什么句子在该语言中是合法的，并指明把词组合成短语和子句的规则。

G 定义为：$G=(T, N, S, P)$

其中，T 是终结符的集合，终结符是指被定义的那个语言的词(或符号)；N 是非终结符的集合，这些符号不能出现在最终生成的句子中，是专门用来描述语法的；S 是起始符，它是集合 N 中的一个成员；P 是一个产生式规则集。

以英语中的一个很小的子集为例，它具有如下的文法：

$G=(T, N, S, P)$

$T=\{$the, dog, cat, runs,…$\}$

$N=\{S$, NP, VP$\}$

$S=\{S\}$

产生式规则集 P 的规则：

$S=$NP$+$VP；$N=$dog；NP$=$the$+N$；$N=$cat；VP$=$runs

根据这些简单的文法，能生成以下两个英语句子：

① The cat runs.　② The dog runs.

Chomsky 希望，如果能找到一种描述英语的形式文法，人们就可以根据它利用计算机“理解”英语。但是到目前为止，这个目标仍然没有实现。从自然语言处理的观点来看，形式地定义一种语言的意义在于：如果系统要处理的句子结构是已知的，那么就比较容易写出一种分析算法对输入语句进行句法分析。

(2) 转移生成语法

1957 年 Chomsky 曾提出转换生成语法(Transformational Generative Grammar)，将句子的结构分为深层结构和表层结构两个层次，并根据形式文法中所使用的规则集不同将语法分为四种类型：无约束短语结构(0 型语法)、上下文有关语法(1 型语法)、上下文无关语法(2 型语法)和正则语法(3 型语法)。一些表达相同意义的句子尽管表层结构不同，但其深层结构却是相同的。例如：

① The car will hit that tree soon.

② That tree will be hit by the car soon.

转换生成语法的原理是通过上下文无关语法生成句子的深层结构(形式语言)，然后应用转换规则再将深层结构转换为表层结构。如果要进行句子分析，则首先要逆向应用转换规则将表层结构转换为深层结构，之后再应用上下文无关语法进行分析。

Chomsky 在语法中完全抛开了语义、语用和语境方面的知识，只局限于一个形式化的机制上，因此很难完全确切地描述自然语言。

(3) 扩展转移网络

1970 年美国的 Woods 根据 Chomsky 创建的转换生成语法，设计了扩展转移网络(Augmented Transition Network，ATN)。

转移网络是自然语言中常用的自动机，每个转移网络由一个状态集和一个标号集组成，它的构成可以表示为：

$$状态\times标号=状态$$

其含义是给定当前状态和当前标号后,可以求得下一步状态.在识别语言时,状态是指当前的句子分析到了哪一步,标号是指当前面临的语法成分是什么。

设有 Q0、Q1、Q2 和 Q3 四个状态。Q0 表示语句分析开始;Q1 表示主语分析完毕;Q2 表示谓语分析完毕;Q3 表示全句分析完毕。

又有 NP 和 VP 两个标号。如图 9.2 所示是一个简单的转移网络,该网络可用来分析许多简单语句,如:

① Mary finished the job.

② Tom plays basketball.

Q0 —NP→ Q1 —VP→ Q2 —NP→ Q3

图 9.2 一个简单的转移网络

自然语言中的句子可能是非常复杂的,如果考虑到句子结构的各种可能性,则转移网络也将变得非常复杂。为了降低复杂性,人们研究如何把转移网络模块化,并尽可能分成层次结构,为此提出了递归转移网络。在递归转移网络中,标号可以是简单的词类,也可以是另一个递归转移网络的名字。

但是递归转移网络仍存在其严重的不足,主要问题是在分析完一个句子之后,它只能给出关于该句子是否符合语法的信息,而不能回答有关该句子的语法结构这类问题。原因在于分析时未能把所得到的信息记录下来。为此,Woods 把递归转移网络加以扩充,成为 ATN。ATN 用一组寄存器存放语法分析信息。它每走一步都要测试一下当前情况并根据测试结果决定做什么动作。最后把各寄存器中的信息综合起来,得到所分析句子的语法结构。

ATN 的弱点在于它对句法的过分依赖,限制了它对语言的处理能力,在某些情况下效率很低。

(4) 格语法

20 世纪 60 年代末,Fillmore 提出了一种新的理论——格语法(Case Grammar)。格语法将自然语言理解中的语法和语义分析结合起来,它的语法规则用于描述语法规律而不是语义规律,但规律所产生的最终结构不是严格表示语法结构而是描述语义关系。

按照 Fillmore 的观点,一个句子可以由情态和命题两部分组成。如果用 S 表示句子(Sentence),用 M 表示情态(Modality),用 P 表示命题(Proposition),则可写为:

$$S = M + P$$

命题 P 是动词与其相关的格。如果用 V 表示动词,用 $C_1,C_2,\cdots,C_n$ 表示各种格,则可写为:

$$P = V + C_1 + C_2 + \cdots + C_n$$

而每一个格又可以表示为一个格标(记为 K)再加上一个名词短语。若用 C_i 表示格,则可写为:

$$C_i = K + \mathrm{NP}$$

K 可以是介词,也可以为空。

情态 M 是一系列从整体上描述句子各方面的术语,主要是指时态、体、形式、方式和时间等,可定义为:

M = Tense，Aspect，Form，Mood，Essence，Modal，Manner，Time

其中

Tense：present，past，future

Aspect：perfect，imperfect

Form：simple，emphatic，progressive

Mood：declarative，interrogative，imperative

Essence：positive，negative，indeterminate

Modal：may，can，must

Manner：adverbial

Time：adverbial

(5) 概念从属理论

1973 年美国的 Schank 提出了概念从属理论(Conceptual Dependency Theory，CD 理论)，这种理论与格语法有相似之处，如句子意义的表达以行为(Action)为中心，并包括一些与其相关的句子其他词的格。但两者之间也存在明显的区别，在 CD 理论中，句子的行为不是由动词表示，而是由原语行为集表示，其中每一个原语是包含动词意义的概念，换言之，行为是由动词的概念表示，而不是由动词本身表示。

让我们看两个例句：

① John gave the vase to Mary.

② Mary received the vase from John.

以上两句尽管侧重点不同、所用的动词不同，但基本概念是相同的，都是关于所有权的转移。但在格语法中，这两个句子的内部存储将采用完全不同的形式。CD 理论指出物体所有权转移是原语行动 ATRANS。因此，上述第一句可表示为：

EVENT1

ACTOR：John

ACTION：ATRANS

OBJECT：the vase

DIRECTION：FROM John TO：Mary

第二句可表示为：

EVENT2

ACTOR：Mary

ACTION：ATRANS

OBJECT：the vase

DIRECTION：FROM：John TO：Mary

EVENT1 和 EVENT2 分别表示了两句的意义，两者的差异仅在于动作的完成者。

由于运用 CD 理论理解自然语言时，大量使用到语义知识，使得对纯粹语法分析有二义性的句子也能赋予唯一的解释。但另一方面，要很好地完成分析工作又需要庞大的语义知识库。

(6) 境况语义学

1983 年美国的 Barwise 和 Perry 建立了境况语义学(Situation Semantics)。境况语义

学是一种语义与语用相结合的语义分析理论。例如：

Tom saw a girl with a telescope.

我们既可以理解成"汤姆用一个望远镜看一个姑娘"，也可以理解成"汤姆看到一个拿着一个望远镜的姑娘"。要能对这句话作出正确的判断，只有根据上下文信息和特定的语言环境。

境况语义学的任务，就是要从语言环境中获取在语法和语义分析中无法得到的信息，更好地完成自然语言理解。

(7) 语料库语言学

近几年来，在国际上掀起了语料库语言学(Corpus Linguistics)的研究热潮。语料库语言学研究机器可读的自然语言文本的采集、存储、检索、统计、语法标注、句法-语义分析以及具有上述功能的语料库在语言定量分析、词(字)典编撰、作品风格分析、自然语言理解和机器翻译等领域的应用。

2) 国内的研究现状

概念层次网络(Hierarchical Network of Concepts，HNC)理论是关于自然语言理解的一个理论体系。它以概念化、层次化、网络化的语义表达为基础，所以称之为概念层次网络理论。HNC 理论把人脑认知结构分为局部和全局两类联想脉络，认为对联想脉络的表达是语言深层(即语言的语义层面)的根本问题。

HNC 理论的中心目标是建立自然语言的表述和处理模式，使计算机能够模拟人脑的语言感知功能。该理论使自然语言理解获得了突破性的进展，它所蕴涵的精深丰富的思想对人工智能、语言学、计算机科学和认知科学等都具有重要的理论和应用价值，对中文信息处理和汉语研究尤其具有实际意义。

HNC 理论完全摆脱了国内现有语法学的束缚，而从语言的深层入手，以语义表达为基础，为汉语理解开辟了一条新路。HNC 理论提出了可供实现的完整的自然语言理解的理论框架，它是一个面向整个自然语言理解的强大而完备的语义描述体系，包括语句处理、句群处理、篇章处理、短时记忆向长时记忆扩展处理、文本自动学习处理。HNC 理论的出发点就是运用两类联想脉络"帮助"计算机理解自然语言。自然语言的词汇是用来表达概念的，因此，HNC 建立的词汇层面的局部联想脉络体现为一个概念表达体系。概念分为抽象概念和具体概念。HNC 理论的概念表达体系侧重于抽象概念的表达。对具体概念采取挂靠近似表达方法。HNC 理论认为应该从多元性表现和内涵两个方面描述概念。

HNC 利用五元组表达抽象概念的多元性，对抽象概念的内涵采用网络层次符号表达。其网络层次符号包含三大语义网络：基元概念语义网络、基本概念语义网络和逻辑概念语义网络。HNC 的五元组符号和三大语义网络的层次符号组合起来就可完成对抽象概念的完整表达，从而为计算机理解自然语言的语义提供了有力手段。

自然语言理解技术大致可分为机器翻译、语义理解及人机会话几个方面。其中机器翻译(Machine Translation，MT)，是利用计算机把一种自然语言转变成另一种自然语言的过程。智能搜索引擎在这一领域的研究将使得用户可以使用母语搜索非母语的网页，并以母语浏览搜索结果。语义理解通过将语言学的研究成果和计算机技术结合在一起，实现了对词语在语义层次上的理解。人机会话技术可以为计算机提供下一代的人机交互接口，实现从文字接口、图形接口到自然语言接口的革命，同时在家用电器的人性化设计方面

有着广泛的应用前景，其技术内涵主要包括语音识别和语音合成两个核心部分。

在语义理解的整个过程中，智能分词技术是最初的一个环节，它将组成语句的核心词提炼出来供语义分析使用。在分词的过程中，如何恰当地提供足够的词供分析程序处理，并且过滤掉冗余的信息，这是后期语义分析的质量和速度的重要前提。尤里卡的智能分词避免了传统分词技术在拆分时产生的歧义组合。从而为语义理解提供了良好的原始材料。同时，在分词过程中，知识库中的同义词会被逐个匹配并同时提交给语义理解模块使用，这样处理过的句子，不仅提供了原始的句型，还同时搭载了语句的概念部分。

4. 关键技术

迄今为止，对自然语言理解尚无统一和权威的定义。按照考虑问题的角度不同而有不同的解释。从微观上看，自然语言理解是指从自然语言到机器（计算机系统）内部之间的一种映射。从宏观上看，自然语言理解是指机器能够执行人类所期望的某些语言功能。这些功能包括：

(1) 回答有关提问；

(2) 提取材料摘要；

(3) 不同词语叙述；

(4) 不同语言翻译。

然而，对自然语言的理解却是一项十分艰难的任务。即使建立一个只能理解只言片语的计算机系统，也是很不容易的。这中间有大量的极为复杂的编码和解码问题。一个能够理解自然语言的计算机系统就像一个人那样需要上下文知识以及根据这些知识和信息进行推理的过程。自然语言不仅有语义、语法和语音问题，而且还存在模糊性等问题。具体地，自然语言理解的困难是由以下三个因素引起的，即：

(1) 目标表示的复杂性；

(2) 映射类型的多样性；

(3) 源表达中各元素间交互程度的差异性。

近几年来，由于以下因素的推动，即：

- 计算机技术的飞速发展；
- 可用的语料库数量的不断增大；
- 经济发展对大量实用处理系统的迫切需要，使语料库语言学的研究得到了迅速的发展。

从 1990 年以来历届重要的国际会议，包括 COLING、ACL 和 TMI 等，每届都有许多新的研究成果出现。而对汉语语料库语言学的研究，近几年来也出了许多研究成果，如自动词性标注、自动分词研究、句法功能标注、语义信息标注、汉语音字转换和汉语语音识别等。但总的来说，发展速度并不是很快，规模也不太大。

5. 应用

采用自然语言理解的智能信息服务创造了电子服务新概念，其最大的特点是具有中文自然语言理解功能。它首先对提出的问题进行断词和断句，然后根据系统预先设置的语义规则理解整句话的意思，形成相应的查询条件，在现有数据库进行快速而准确查询，给出用户需要的答案或者提示用户进一步输入有关的信息。由于自然语言理解技术发展的阶段性，目前还做不到像人一样具有足够智能的、通用的理解，但是只要划定具体的领域（领域范围可大可小，如天气、体育等简单信息查询，也可以是交友、购物和保健等宽泛的智能聊天

或咨询等),性能先进的自然语言理解技术都能量身定做出具有足够实用性的智能信息服务系统,自动理解客户用自然语言发出的相关领域的问题,使网络交流更人性化,信息查询更方便、快速和准确,从而获得高质量的服务。

目前,自然语言理解的应用主要涉及以下几个方面。

1) 智能短信服务

短信服务商可根据具体应用领域定制一系列的智能短信业务,如旅游交通、金融证券、交友网聚和智力竞猜等受欢迎的服务,让服务商和用户都抛开厚重的手册和复杂的编码,代之以口语化的自然语言通过短信输入,短信系统能迅速理解用户的意图,提供准确、周到的信息和服务。这样的服务将直接刺激用户使用短信服务的兴趣和频率,并为短信服务商和电信运营商开发更新的、更具吸引力的业务创造良好条件。

2) 智能聊天机器人

目前流行于各大网站、各类即时通信软件的聊天机器人已经应用成熟的自然语言理解技术,不仅实现了基本的、简单的对话交流、信息查询等功能。还可以通过与用户对话、聊天等生动、灵活的形式,了解用户的需求,利用智能搜索技术采集和分析互联网和知识库的内容,进行自动过滤、筛选、获取有效内容,对相关信息内容进行智能化编辑整理,并返回给用户;这样,聊天机器人的服务将会更易用和实用,能够实现功能强大而实用的智能聊天、智能游戏、个性化的新闻定制、智能网络搜索、智能电子商务等,给用户带来极大的方便和无穷的乐趣。

3) 智能搜索引擎

普通的搜索引擎引入自然语言理解和知识管理技术,构造成新的智能搜索引擎,就能提供全新的信息查询服务,创造更综合的增值服务。目前的搜索引擎由于只使用关键词匹配,没有引入自然语言理解,每次搜索时只是按照关键词进行匹配,返回的大量信息和链接,其中很大部分是垃圾信息或者不是用户需要的信息,往往导致用户无所适从。而门户网站或者专业网站虽然对信息进行很好的分类,但是首先要用户记住网址,还要懂得分类标准,然后逐层单击相应的分类链接,才能获得所需信息。这样的信息分类查询给信息服务商带来很大的工作量,服务成本和进入门槛显著提高,而用户使用起来也不方便,导致用户的流失。基于内容的智能搜索引擎是依靠语义网络、分词、句法分析、同义词处理等语言理解技术最大程度地了解用户的信息需求,获得更高的易用性、更准确的范围定位和更智能的搜索结果。

运用先进的自然语言理解技术,智能搜索引擎可以识别并回答用户的问题,使用户摆脱传统搜索引擎基于关键字的束缚,指引用户更有效、更快捷地寻找到所需信息,同时为用户提供相关的、有参考价值的其他内容。由于这些特点,使得智能搜索能够在互联网信息检索的各个方面得到广泛应用。它可以为大型综合搜索引擎提供后台支持,使之具有人性化、交互性的特点。它能够方便地实现垂直搜索引擎的专业类别搜索,当然也可以为信息门户网站提供方便快捷的站内信息搜索服务。

智能搜索引擎除了在互联网上使用外,也可以支持 WAP 协议而应用在手机上。其实用户往往在逛街、旅游、交通等室外环境、移动状态时更需要随时查询信息。手机是很好的查询工具,但是手机的屏幕小、内存少、带宽窄,不适合接收和保存大量的信息,更不可能翻看数以百计、千计的信息。传统的基于关键词搜索或者分类信息查询模式都存在一定的缺

陷,特别不适合在手机上使用;而基于自然语言理解的智能搜索引擎,能很好地为手机用户提供随时、随地、随心的信息服务。

用户们呼唤自然语言理解,因为它是人机交互的最高境界,是人机交互最自然的方式。有理由相信,在不远的将来,任何人、任何时刻、在任何场所和任何设备上,都可以通过自然语言方便地浏览网页、互递信息,实现随时随地沟通交流的目标。

9.1.3 实现过程

与结构化的数据相比,Web 是异构的、非结构化或半结构化的、动态的数据,并且容易造成混淆,所以很难直接实现 Web 挖掘,必须经过必要的预处理。

典型的 Web 挖掘实现过程概括如下:

(1) Web 资源搜集。

Web 资源搜集旨在获取数据,值得注意的是有时信息资源不仅限于在线 Web 文档,还包括电子邮件、电子文档、新闻组或者网站日志甚至是通过 Web 形成的交易数据库的数据。

(2) 预处理。

预处理是指从获取的 Web 资源中剔除无用信息并进行必要的处理,例如从 Web 中自动删除广告链接、多余格式标记、自动识别段落或者字段并组织成规整的逻辑形式甚至是关系表。

Web 的内容是人类所使用的自然语言,计算机很难理解其语义。这些特殊性使得现有的数据挖掘技术无法直接应用于 Web 挖掘。需要对 Web 进行分析,抽取代表其特征的元数据。这些特征可以用结构化的形式保存,作为 Web 的中间表示形式。

在对 Web 进行特征提取前,需要先进行预处理。对于英文需进行词干化(Stemming)处理,中文的情况则不同,因为中文的词与词之间没有固有的间隔符(空格),需要进行分词处理。所谓分词是在中文文本的各词条间加入分隔符,将中文文本的连续字流形式转化为离散的词流形式。引入分词主要是为后继的处理做准备。自 20 世纪 80 年代初提出自动分词以来,已出现许多分词方法,目前主要包括正向、逆向最大匹配法、逐词遍历法、最佳匹配法和词频统计法,此外还有二次扫描法和邻接约束法等。大致可以归纳为四类:第一类为基于词典的机械分词算法;第二类为基于统计的分词算法;第三类为第一、二类混合的分词算法;第四类为基于知识的分词专家系统。在具体应用中,需要根据具体情况选择不同的分词方法,不同分词方法的正确性很大程度上取决于所建的词库。一个词库应具有完备性和完全性两个方面。所谓完备性,简单而言是对任意一个字串,总能按词库找到对它进行切分的方法;所谓完全性,则意味着词库应当包含所有的词。建立一个同时满足上述两个要求的词库具有很大难度,而对于某一系统而言,可能只用到其中的一部分,因此在构造词典时需要量力而行,在完备和效率之间折中。

(3) 特征抽取。

(4) 模式发现。

自动地发现模式,Web 挖掘所产生的知识模式,既可以是对各 Web 含义的概括,也可以是有关整个 Web 集合的结构或趋势描述。

(5) 模式评价

验证、解释上一步骤产生的模式,既可以机器自动完成,也可与分析人员一起完成。

最后对挖掘出的模式进行质量评价,若评价的结果满足一定的要求,则存储这一知识,否则返回到前面的某个步骤,分析改进后进行新一轮的挖掘。

值得注意的是,Web挖掘作为一个完整的过程,在挖掘之前的信息检索(Information Retrieval,IR)和信息抽取(Information Extraction,IE)相当重要。IR旨在获取相关的Web;IE旨在从Web中找到所需信息,对文档的结构和所表达的含义感兴趣,其重要任务之一是对Web进行组织、整理并适当建立索引。

IR和IE的研究由来已久,随着Web技术的发展,基于Web的信息检索和抽取获得更多的关注。由于Web数量庞大,且动态变化,以原来手工方式收集Web早已力不从心,目前采用自动化、半自动化的方法实现Web信息检索和抽取。

综上所述,Web挖掘的基本流程如图9.3所示。

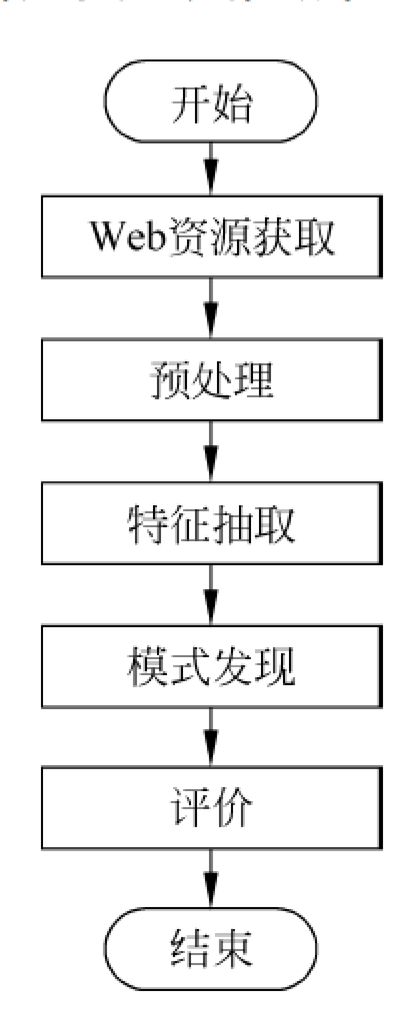


图9.3 Web挖掘的基本流程

对Web的处理可分为数据获取、数据预处理、数据的转换集成、知识发现和模式分析等几个步骤,如图9.4所示。

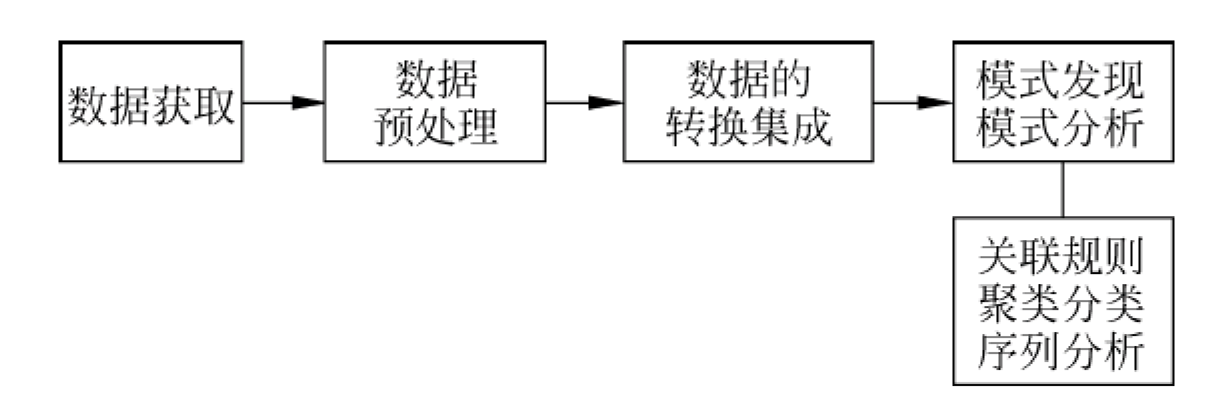


图9.4 Web数据处理步骤

数据来源主要包括Log日志,记录了用户的访问页面、时间和IP地址等主要信息;Web内容,用户浏览的文字、图片等;用户注册信息,Web站点采集的用户输入的自身信息;Web结构数据,指Web本身在频道和链接上的布局。获取的方法主要有服务器端信息,服务器端的日志文件是Web使用挖掘重要的数据来源;客户端的数据收集,用户客户端的Log文件记录了该用户对各个网站的访问情况,比起服务器端的日志能更准确地记录用户的访问行为,但是由于隐私保密,需要用户同意才能获得;代理服务器端的数据收集,代理服务器端的日志文件记录了客户端用户通过代理服务器对各个网站的访问行为,但是由于cache的大量存在,使得代理服务器端的Log数据不能准确地确定用户和时间,采集信息也不全面。

Web的预处理包含数据清洗、用户识别、会话识别和事务识别等过程。数据清洗是指删除Web日志中与挖掘无关的数据,目的是消除冗余;用户识别指分离特定用户的访问日志,常用的方法主要有通过用户注册信息,利用cookie技术和利用Web站点结构结合启发式的规则确定用户;会话识别的目的是将用户的访问记录分割成若干个独立的会话进程,来挖掘会话间的知识信息;事务识别用来细化会话的粒度,采用的方法主要有引用长度、最大向前路径和时间窗口。前两种方法是基于语义对用户会话进行划分的,后一种方法通常

作为比较其他两种方法的基准。

数据的转换和集成是把经过预处理的数据,按照某种格式统一存储到数据库中。

知识发现和模式分析是利用格式化的数据进行关联分析、序列模式发现、访问路径分析和分类聚类分析等。知识发现和模式分析取决于最终的挖掘目标,通过挖掘,发现对决策用意义的信息。上述步骤是 Web 挖掘任务的实施过程,需要对客户的需求有明确的理解,以及对挖掘结果做出合理的解释和展现。

9.2 Web 资源获取

1. 定义

网络爬虫(Crawler)是一种自动提取网页的程序,使搜索引擎从互联网上下载网页,是搜索引擎的重要组成部分。爬虫从一个或若干初始网页的 URL 开始,通过分析该 URL 的源文件,提取出新的网页链接,继而通过这些链接继续寻找新的链接,这样一直循环下去,直到抓取并分析完所有的网页为止。当然这是理想状态下的爬虫,但实际上要抓取互联网上所有的网页是不可能的,从目前公布的数据来看,最好的搜索引擎也只是获取了整个网络上大约 40%的网页。这有两个原因:其一是网络爬虫的抓取技术瓶颈造成的,无法遍历所有的网页,很多网页链接不能从其他网页上得到;其二是存储技术和处理技术造成的,如果按每个页面是平均 20KB,那么 100 亿个页面大小就是 200 000GB,现有的存储技术根本无法满足。

2. 常用工具

1) Nutch

Nutch 是一个开源的 Java 实现的搜索引擎。它提供了运行搜索引擎所需的全部工具。

Nutch 是一个基于 Lucene,类似 Google 的完整网络搜索引擎解决方案,基于 Hadoop 的分布式处理保证了系统的性能,类似 Eclipse 的插件机制保证了系统的可客户化,而且很容易集成到应用中。

相对于商用搜索引擎,Nutch 具有下列优势:

(1) 透明度高。商用搜索引擎都采用私有的排序算法,因此无法知道搜索出来的排序结果是如何算出来的。而 Nutch 的开源让任何人都可以查看其排序算法是如何工作的。另一些搜索引擎允许竞价排名,而不是根据它们本身的价值进行排序,这样的索引结果并不是和站点内容相关。对学术搜索和政府类站点的搜索而言,一个公平的排序结果是非常重要的,因此 Nutch 是一个好选择,更值得大家信赖。

(2) 良好的扩展性。集成 Nutch 的方式十分灵活。它可以直接集成到站点,为用户提供搜索服务;采用 Nutch 的插件机制打造一个搜索不同信息载体的搜索平台;还可以被定制并集成到应用程序中。

(3) 开放的源代码让我们可以了解一个大型分布式的搜索引擎是如何工作的,从而对搜索引擎有更好的理解。

2）Heritrix

Heritrix 是一个开源、可扩展的 Web 爬虫。Heritrix 设计成严格按照 robots.txt 文件格式和 META robots 标签。

Heritrix 是一个爬虫框架，可加入一些可互换的组件。它的执行是递归进行的，主要步骤如下：

- 在预定的 URI 中选择一个。
- 获取 URL。
- 分析，归档结果。
- 选择已经发现的感兴趣的 URI，加入预定队列。
- 标记已经处理过的 URL。

Heritrix 主要包括三大部件：范围部件、边界部件和处理器链。

(1) 范围部件：主要按照规则决定将哪个 URI 入队。

(2) 边界部件：跟踪哪个预定的 URI 将被收集，和已经被收集的 URI，选择下一个 URI，剔除已经处理过的 URI。

(3) 处理器链：包含若干处理器获取 URI，分析结果，将它们传回给边界部件。

3）WebSPHINX

WebSPHINX 是一个 Java 类包和 Web 爬虫的交互式开发环境。Web 爬虫（也叫做机器人或蜘蛛）是可以自动浏览与处理 Web 页面的程序。WebSPHINX 由两部分组成：爬虫工作平台和 WebSPHINX 类包。

4）WebLech

WebLech 是一个功能强大的 Web 站点下载和镜像工具。它支持按功能需求下载 Web 站点并能够尽可能模仿标准 Web 浏览器的行为。WebLech 具有一个功能控制台并采用多线程操作。

5）Arale

Arale 主要为个人使用而设计，而没有像其他爬虫那样关注于页面索引。Arale 能够下载整个 Web 站点或来自 Web 站点的某些资源。Arale 还能够把动态页面映射成静态页面。

6）J-Spider

J-Spider 是一个完全可配置和定制的 Web Spider 引擎。可以利用它来检查网站的错误（内在的服务器错误等）和网站内外部链接，分析网站的结构（可创建一个网站地图），下载整个 Web 站点，还可以写一个 JSpider 插件扩展所需要的功能。

7）Spindle

Spindle 是一个构建在 Lucene 工具包之上的 Web 索引/搜索工具。它包括一个用于创建索引的 HTTP Spider 和一个用于搜索这些索引的搜索类。Spindle 提供了一组 JSP 标签库使得那些基于 JSP 的站点不需要开发任何 Java 类就能够增加搜索功能。

8）Arachnid

Arachnid 是一个基于 Java 的 Web Spider 框架。它包含一个简单的 HTML 解析器能够分析包含 HTML 内容的输入流。通过实现 Arachnid 的子类就能够开发一个简单的 Web Spiders 并能够在 Web 站点上的每个页面被解析之后增加几行代码调用。Arachnid 的下载包中包含两个 Spider 应用程序例子用于演示如何使用该框架。

9）LARM

LARM 能够为 Jakarta Lucene 搜索引擎框架的用户提供一个纯 Java 的搜索解决方案。它包含能够为文件，数据库表建立索引的方法和为 Web 站点创建索引的爬虫。

10）JoBo

JoBo 是一个用于下载整个 Web 站点的简单工具。它本质是一个 Web Spider。与其他下载工具相比其主要优势是能够自动填充 form（如自动登录）和使用 cookies 处理 session。JoBo 还有灵活的下载规则（如通过网页的 URL、大小、MIME 类型等）来限制下载。

11）snoics-reptile

snoics- reptile 是用纯 Java 开发的，是进行网站镜像抓取的工具，可以使用配制文件中提供的 URL 入口，把这个网站所有的能用浏览器通过 GET 的方式获取的资源全部抓取到本地，包括网页和各种类型的文件，如图片、flash、mp3、zip、rar、exe 等文件。可以将整个网站完整地下载至硬盘内，并能保持原有的网站结构不变。只需要把抓取下来的网站放到 Web 服务器（如 Apache）中，就可以实现完整的网站镜像。

12）Web-Harvest

Web-Harvest 是一个 Java 开源的 Web 数据抽取工具。它能够收集指定的 Web 页面并从这些页面中提取有用数据。Web-Harvest 主要是运用了像 XSLT、XQuery 正则表达式等技术实现对 text/xml 的操作。

9.3 Web 预处理

Web 挖掘与传统的数据挖掘相比具有许多独特之处。首先，Web 挖掘的对象是大量、异质和分布的 Web。我们认为，以 Web 作为中间件对数据（仓）库进行挖掘，以及对 Web 服务器上的日志、用户信息等开展挖掘，仍属于传统的数据挖掘范畴；其次，Web 在逻辑上是一个由文档结点和超链接构成的图，因此 Web 挖掘所得到的模式可能是关于 Web 内容的，也可能是关于 Web 结构的。此外，由于 Web 本身是半结构或无结构的，且缺乏机器可理解的语义，而数据挖掘的对象局限于数据（仓）库中的结构化数据，并利用关系表等存储结构发现知识，因此有些数据挖掘技术并不适用于 Web 挖掘，即使可用也需要建立在 Web 预处理的基础之上。这样，开发新的 Web 挖掘技术，以及对 Web 进行预处理以得到关于 Web 的特征表示，便成为 Web 挖掘研究的重点。

9.3.1 Web 过滤

浏览 Web，我们会发现它们通常包含两部分内容：一部分内容体现的是网页的主题信息，例如一个新闻网页中的新闻部分，称为“主题”内容；另一部分则是与主题内容无关的导航条、广告信息、版权信息以及调查问卷等内容，称为“噪音”内容。噪音内容通常分布在主题内容周围，有时也夹杂在主题内容中间，通常是以链接导航文字（anchor text）的形式出现。噪音内容一般无内容相关性，同时，它们链接的网页常常也无内容相关性。这样，网页中的噪音不仅给基于网页内容的应用系统带来困难，也给基于网页超链接指向的应用系统带来麻烦。

1. 国内外研究现状

随着网络技术的发展,基于 Web 的信息检索变得越来越重要。但是 Web 除了有用的信息之外,还包括一些和网页主题内容关联不大的部分,如网页的页眉、导航栏和著作权声明等,这些部分被称作噪音块。为了保证网页信息检索的准确性,在对网页进行关键词抽取或主题抽取等操作前,需要将噪音块过滤出去。如何采用自动方式去除网页噪音成为研究的一个热点。

有文献认为所有的噪音块具有相似的风格,基于此提出了 Site Style Tree(SST,站点风格树)的思想,在对网站网页取样建立 SST 后,通过 SST 的测量,得到噪音部分。这种算法的目的是在整个网站中寻找噪音部分。

Yu 等提出了基于视觉的 Web 页面分割(Vision-based Page Segmentation,VIPS),该算法综合考虑了文件对象模型(DOM)中具有特殊作用的标签(如 P 表示一段文字的开始,UL 用来表示一个列表)和视觉上的分隔符(如空白区域、字体大小),将一个网页分割成语义上相近的段落。

有文献针对动态产生的网页使用 Hish 算法对 Augmented Fragment(AF)进行 shingles 编码,通过编码检测网页子树的变化情况,由此找出噪音部分。上述算法能够找到网页中的重要部分,但是那些不重要的部分不一定为噪音块。

在对网页进行分类时,由于噪音内容与主题无关,因此训练集中的噪音内容会导致各类别的特征不够明显,而待分类网页中的噪音内容则会导致该网页类别不明确,因而影响网页自动分类的效果。将网页净化用于分类器的目的则是先去除原始网页中的噪音内容,而后,利用净化后的网页进行分类器训练(特征项选取)并对净化后的网页进行分类,从而达到提高分类准确性的效果。

在主题搜索领域,大量的广告、导航条等噪音内容会导致主题漂移(topic drift)。这说明传统的主题搜索算法中以网页为粒度构造的网络图不够准确,必须深入到网页内部将处理单元的粒度缩小,才能提高内容分析的准确性。在网页信息提取领域,自动识别模式的方法必须要从整个网页中提取模式,而不是只针对主题内容进行提取。因此,在净化后的网页上作信息提取不仅可以排除噪音信息对信息提取的干扰,提高信息提取的准确性,而且可以使得网页中的结构简单化,提高信息提取的效率。

从上述分析看到,网页中有用的大量数据往往都被许多噪音数据所干扰,包括广告、导航条、版权说明等。尽管这些噪音对于在互联网上浏览的用户而言有一定的功能性作用。但是,它们也妨碍了网页的自动收集和挖掘,包括网页自动分类、聚类、信息抽取和信息检索等的准确性、效率和性能。噪音对基于 Web 内容的智能搜索平台的研究的影响是普遍而严重的,而通过比较去除噪声前后网页的分类正确率,可以看出使用净化后的网页进行分类器的训练和分类,会较好地提高分类精度,同样,将净化后的网页应用到基于内容的网页文本处理系统中,也会很好地提高网页处理系统的精度,所以网页净化成为 Web 预处理环节中一个必不可少的工作。

2. 主要算法

目前,比较通用的网页噪音过滤算法如下:

(1) 基于可视化布局信息的网页噪音过滤算法。针对单一网页中包含的丰富的可视化

信息，获取页面中各个元素标记的布局信息，然后利用布局信息对页面进行划分，最后在此基础上去除噪音结点。

(2) 基于网页内容结构表示法过滤噪音算法。多数网站的 Web 是由预先定义好的模板(templates)自动生成，具有相同的布局结构和风格。如果网页中部分结构和内容(如图像、链接等)在多个 Web 中出现，这些部分就是噪音；相反，仅在少数网页中出现的结构和内容，就是要提取的主题内容。因此，基于网页内容结构表示法过滤噪音算法的净化过程可以分为两个步骤：网页内容结构的表示和网页内容块的取舍。

(3) 基于同层网页相似性过滤网页噪音算法。该算法是基于同层网页具有相似结构这一前提去除网页噪音。同层网页即在网站的导航结构下同属于同一个结点的子结点的网页。现在的网站设计多采用自动生成程序(如 PHP、JSP)或模板，这样导致大部分网站的同层网页具有类似的显示效果，而这些在同层网页中多次重复的模块大多是导航栏、著作权声明等噪音。通过网络爬虫分析目标网页中的链接可以获取同层网页。

(4) 基于模糊理论的网页过滤算法。基于网页内容进行分析，对内容块中的信息进行关键字分析，通过关键字和数据库中的过滤字典进行内容分析和识别，如果内容块中包含的“垃圾”词汇大于总词汇的一个阈值，则可以判别该区域块为噪音区域。

(5) 基于网页主题内容识别的噪声过滤算法。针对一个网面而言，字数最多的文本区域块便是该网页文件所要表现的内容，其余的内容块有的是和主题相关的背景介绍文字，或者是相关链接等，它和其他内容不相关的广告链接，导航区域以及页面版权信息分布在主体内容周围。该算法主要是通过识别网页的主题内容块，然后再判别其余内容块和主题内容块的相关度，和主体内容块相似度低的内容块是噪音块。

(6) 基于网页结点特征识别和判定过滤噪音结点算法。同样的，网页可以分成几个内容区域，依据每一个内容块中所包含的词项数与图片数和超链接数的比值可以为每个内容块设定一个类型，分为主题型、多链接型和图片型三种。如果内容块中词项数与图片数的比值小于某一阈值，该内容块就是图片型；如果内容块中作为链接导航文字出现的词项数与该区域中总词项数的比值小于某一阈值，该内容块就是多链接型，否则为主题类型。这样，标签树中每个结点都有类型和属性集两组描述性信息，以及超链接集和重要标签集等数据信息。通过每个结点的属性，便能判定它们属于哪一个特征集。

上述六种主要的网页去噪算法从不同的角度作为解决问题的切入点，从不同的侧重点进行实现，比较算法实现的效率和实际过滤的效果，它们都各有优劣。但是 Web 2.0 时代的到来，AJAX 技术的大量运用，DIV 和 CSS 结合新的标准布局，Widget 技术的异军突起，都使得网页噪音以各种新形态出现，使得网页去噪技术有着日新月异的改进，广告展示和网页去噪技术的较量是一场没有硝烟的战争。

快速准确地识别并清除网页的噪音内容(称为网页净化)是提高 Web 应用程序处理准确性的一项关键技术。首先，网页净化后，没有了噪音的干扰，Web 应用程序可以以网页的主题内容为处理对象，从而提高处理结果的准确性。其次，网页净化可以显著简化网页内标签结构的复杂性并减小网页的大小，从而节省后续处理的时间和空间开销。因此，网页净化已成为 Web 信息系统预处理环节中一个必不可少的工作。

1）基于可视化布局信息的网页过滤算法

虽然XML发展很快，但目前网络上大部分的页面还是由HTML实现，但是由于HTML本身不具备自描述的特性，页面书写时负责显示和承担主题描述的信息混在一起，并且设计者可以随意的把各种内容加入到页面中，因此网页中充满与主题无关的噪音是常见的现象。网页噪音根据其划分粒度的不同，可从整个Web和单一页面本身加以定义。

定义1：全局噪音

在对Web上的一组页面集进行挖掘或聚类时，若一个网页所存留的副本，如镜像网站、复制的页面及旧版本的页面都在此页面集中，则这些副本称为噪音，即全局噪音。

定义2：局部噪音

在一个页面内与页面主题无关的区域和项。这些噪音包括广告栏、导航条和修饰作用的图片等。

Web中常见噪音的类型如下：

- 导航类——为了维持网页间的链接关系，方便浏览者对网站进行浏览而设置的链接。
- 修饰类——为了美化页面而采用的背景、修饰图片和动画等。如站点标志图片、广告条。
- 交互类——为了收集用户提交信息或提供站内搜索服务的表单等。如在线的问卷调查表。
- 其他类——网页中声明的版权信息、创建时间和作者等描述性信息。

在对网页的编程中会用到文档对象模型(DOM)。DOM提供了访问页面中各个元素属性与方法的接口。每一个网页都会对应一个DOM树。通过对DOM树的遍历，可以对页面中每一个元素进行处理。如可访问元素所对应位置的坐标，元素所占空间的宽度和高度等。如图9.5所示为百度首页的DOM树，其中每个结点都有字节相应的属性。

```
table width="600" cellspacing="0" cellpadding="0" border="0">
 ⊟ <tbody>
    ⊟ <tr valign="top">
        <td width="92"/>
      ⊟ <td height="62">
         ⊟ <form action="http://www.baidu.com/s" name="f">
              <input class="kw" type="text" maxlength="100" size="36" name=
            ⊞ <script>
              <input type="hidden" value="3" name="cl"/>
              <input class="sb" type="submit" value="百度一下"/>
              <br/>
              <br/>
              <input type="hidden" value="mozillacncb" name="tn"/>
           </form>
        </td>
```

图9.5　百度首页的DOM树

算法思想：

虽然网页中的噪音很多，且网页设计没有统一规范，但设计者大部分都将噪音信息放在页面中的次要位置，以突出主题。如版权信息一般在页面底部，广告栏一般在顶部等。这种设计习惯也为利用布局信息去除噪音带来了便利。

噪音过滤算法的原理基于以下常识：

（1）页面设计布局上的常规做法

通常设计者在设计页面时会先进行布局的设计，即先用表格或图层等标记在页面显示区域中勾画出页面内容分布的区域。然后再在特定区域内进行细化的页面设计，把所需元素标记放进去。这种设计方法，会用一些区域来存放导航条、版权、广告栏等信息，即这些区域为噪音区域。

（2）用户在浏览页面时识别页面主题的常识

用户在浏览网页时，可以很容易地找到页面中表达主题的区域而不会被导航条和广告栏等所迷惑。这是因为用户会将自己的注意力集中于页面中部大块的区域，即设计者通常用来存放页面主题内容的区域。用户浏览时会下意识地将页面进行划分，而将一些噪音区域略掉，这正是机器去除噪音难于做到的事情。

如图 9.6 所示，在以上常识基础上，可将页面布局粗分成如下五个区域：HEADER、FOOTER、LEFT、RIGHT、CENTER，分别代表页面的上、下、左、右及中部区域。通过阈值可调整各区域大小。其中 H 代表 HEADER，F 代表 FOOTER，LM 和 RM 分别代表左和右两边区域，C 代表中部区域。H_1、H_2、W_1、W_2 分别代表区域划分时所用的阈值。

图 9.6　可视化页面的区域结构图

算法：VL_PS 网页噪音去除
输入：任一个网页文件
输出：去除噪音后的网页
将输入网页转化为 DOM 结构，并显示，由 DOM 得到页面实际大小
根据页面实际大小决定各区域的阈值 H_1（上边界），H_2（下边界），W_1（左边界），W_2（右边界）
对页面内所有元素（BODY 标记除外）取得其绝对坐标及实际大小（宽度和高度）和由阈值所划分出的区域进行比较
若任一个元素 E，其布局信息记为（LEFT，TOP）为其所占区域左上角原点绝对坐标.（WIDTH，HEIGHT）记为该元素所占区域大小.则：
IF E.TOP + E.HEIGHT <= H_1，则 E 属于 HEADER；
IF E.TOP >= H_2，则 E 属于 FOOTER；
IF E.LEFT + E.WIDTH <= W_1，则 E 属于 LM；
IF E.LEFT >= W_2，则 E 属于 RM；

所有未被划分出去的元素属于 CENTER，将处于 CENTER 中的页面内容作为结果返回。

上述算法通过一次对 DOM 树的遍历即可实现页面区域的划分，并去除噪音。算法中各元素布局信息的计算，均采用了元素在页面中所处的绝对坐标。由于在 DOM 树中各元素存在层次关系，因此由 DOM 直接得到的坐标信息是相对于其父元素的，为得到元素的绝对坐标，可通过回溯由元素到 BODY 标记这样一条元素到根的 DOM 树的路径，叠加各坐标值获得。

该算法最大的特点是对主题内容定义清晰的网页处理准确率非常高，而且处理的速度

也比较快,但是对于主题内容中夹杂噪音的情况就不容易识别,而且对于文字内容比较少的,还会出现误处理的情况。通过对5000篇新浪和搜狐网页的分析,在实验中设定上边界 H_1 为200,下边界 H_2 为160,W_1 和 W_2 为页面实际宽度的30%。实验中将分块及去除噪音结果分为好、较好、较差和差四个等级,在每级中包含的页面数占比如表9.1所示。

表9.1 页面集噪音去除结果

好	较好	较差	差
24%	52%	22%	2%

综上所述,基于可视化布局信息的网页噪音去除的算法思想非常明确和适用,能够处理大部分的网页结构,对于主题内容区域明显的网页处理效果较好,但对于噪音嵌入主题区域块的网页的效果不甚理想。

2) 基于网页结点特征识别的网页过滤算法

HTML是一种标识语言(Markup Language),它定义了一套标签刻画网页显示时的页面布局。因此,对于HTML网页最常用的结构表示方法是构造网页的标签树。现有的标签树构造工具很多。DOM是常用的标签树构造工具,它可以将网页中的标签按照嵌套关系整理成一棵树状结构。针对网页净化的特殊需求,首先对HTML中的标签按照功能进行分类,进而提出更加适合网页净化的标签树的构造方法。依据标签的作用可以将HTML标签分为两类:

(1)规划网页布局的标签。如上所述,在视觉上,网页是由若干内容块组成的,而内容块是由特定的标签(称之为容器标签)规划出的。常用的容器标签有<table>、<tr>、<td>、<p>和<div>等。

(2)描述显示特点的标签。除了描述布局结构的标签外,HTML标准中还定义了一套标签描述其包含的内容本身。例如,<b>标签说明所包含的内容要用粗体显示,<img>标签说明包含的是一个图片,<a>标签说明是一个超链接等。

由于网页净化是以内容块为单位进行保留和删除的,因此,依据容器标签构造标签树中的结点较为合理。而其他类型的标签信息可以作为它所在的内容块的属性存在。网页的源文件如图9.7所示,一棵完整的标签树如图9.8所示,其中link-list和weighty-tag-list分别记录着网页中超链接信息和重要标签信息(如<b>、<h1>等)。标签树构造完成后,网页净化过程就变为对标签树中结点的剪裁。除了上述基本的标签信息外,我们还可以归纳计算出以下描述性信息。

Web根据内容可分为主题网页、目录型网页和图片网页三类。三种网页的净化方法各不相同。本节首先描述三种网页的特征和判别方法,然后对面向主题网页的净化算法进行详细介绍,最后简要介绍面向目录型网页和图片网页的算法。

网页类型判别方法从内容上看,三种类型的网页有着较为明显的特征。主题网页通常通过成段的文字描述一个或多个主题,虽然也会有图片和超链接,但这些图片和超链接并不是网页的主体。图片网页中内容是通过图片体现的,而文字仅仅是对图片的一个说明,因而文字不多,这也导致传统的向量表示方法对图片网页不适用。目录型网页通常不会描述一件

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.o
<HTML xmlns="http://www.w3.org/1999/xhtml">
<HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>阳光地带-IsMayDay</title>
</HEAD>
<BODY>
<div id="wrapper" class="r1_a">
    <div id="header" style="width:1000px;height:90px;font-size:9pt;"></div>
    <div id="center" class="d1"  style="color:#ccc;margin:5px;">
        <div id="left" style="color:#ccc;margin:5px;">
            <ul>
                <li>设置</li>
                <li>处理</li>
                <li>结果</li>
            </ul>
        </div>
        <div id="main">
            <div id="content_1">网页过滤噪音系统,这是一个</div>
            <div id="content_2">由于网页净化是以内容块为单位进行保留和删除</d
        </div>
    </div>
    <div id="footer"></div>
</div>
</BODY
```

图 9.7　网页的源文件

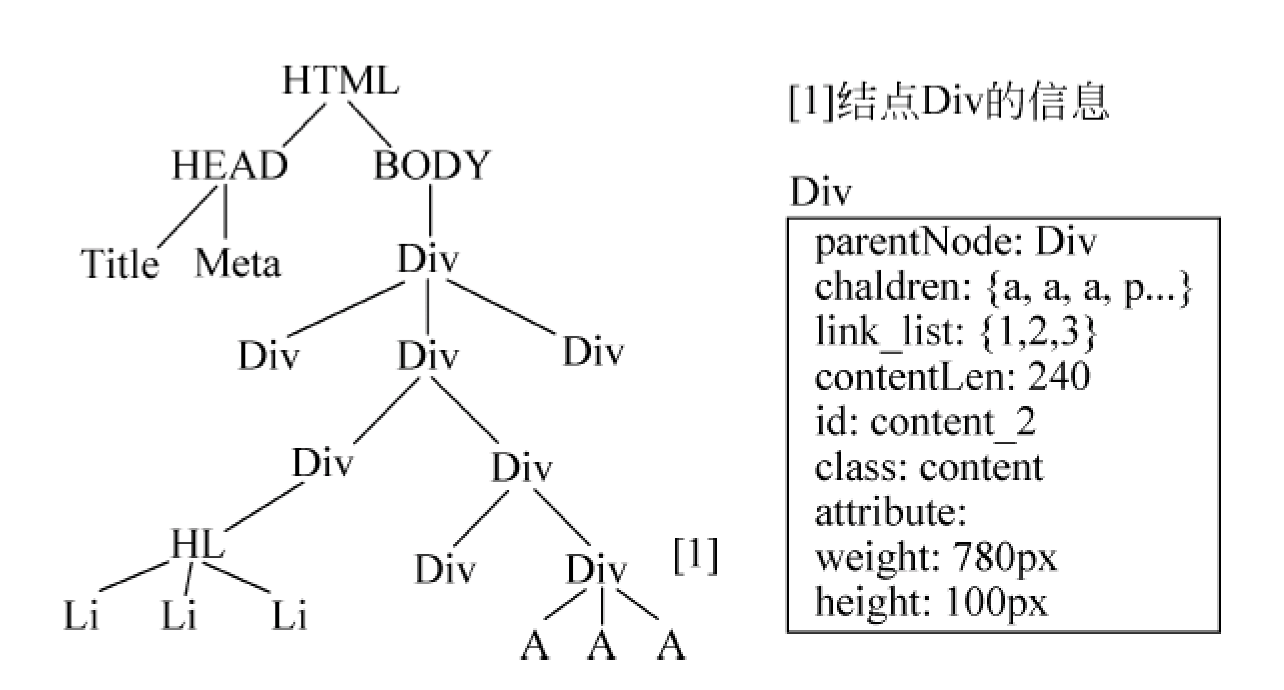


图 9.8　对应的 Dom 解析树和[1]结点的信息表示

事物,而是提供指向相关网页的超链接,因此,网页中超链接密集。

网页都有一定的布局,比如分左右两边或是中间和边缘。网页作者通常将重要的内容放在网页的中间部分,而边缘部分内容的重要性相对较低,这也符合人的浏览习惯。因此,依据网页中间区域的内容判断网页的类型是相对合理的,而网页中内容的位置信息在构造的标签树中是通过内容块的属性记录下来的。在构造标签树时,依据内容块中词项数与图片数的比值以及内容块中词项数与超链接导航文字中词项数的比值将网页中的内容块分为主题型、多链接型和图片型三种,对于基于内容块的类型,同样可以使用网页中间区域多链接型内容块包含的词项数与网页中间区域词项数的比值来判断网页是否为目录型网页。同理,使用网页中间区域图片型内容块包含的词项数与网页中间区域词项数的比值可以判断网页是否为图片网页。

网页净化过程是在网页内部以内容块为单位进行类型鉴别,以及重要性和相关性评价。使用如式(9.1)所示的特征项权值计算公式,其中 BWeight 表示内容块的权值,其值由一个内容块中的重要标签来决定;BN 为网页中内容块的总数;n 为一个内容区域中超链接的总数,$BWeight_j$ 为网页中内容块的权重;BTf_{ij} 为关键词 i 出现在内容块 j 中的词频。

$$w_i = \frac{\sum_{i=1}^{BN} \text{BWeight}_j * \text{BTf}_{ij}}{\sqrt{\sum_{j=1}^{n}\left(\sum_{i=1}^{BN} \text{BWeight}_j * \text{BTf}_{ij}\right)^2}} \tag{9.1}$$

从上述净化方法中可以看到,网页的内容分为主题内容、主题相关内容和噪音内容三部分。其中主题相关内容可以用来调节净化力度,在不同需求的应用中,可以保留也可以去除。在网页分类中,保留了主题相关内容,而在网页去重中,则没有保留。通过实验表明,在保留主题相关内容的净化力度下,净化后的网页平均大小为原始网页的59%,而不保留的情况下,只有原始网页的51%。

9.3.2 Web 去重

Internet 上超过 300 亿个海量网页使互联网成为当今世界上最大的信息库和全球范围内传播信息的最主要渠道,也为广大用户提供了一个价值巨大的信息获取渠道。

网页的规模反映了互联网的内容丰富程度。自 2003 年开始,中国的网页规模基本保持翻番增长,2009 年网页数量达到 336 亿个,年增长率超过 100%。图 9.9 是中国第二十五次互联网发展报告提供的关于 2003—2009 年的网页数统计和增长率情况。从图 9.9 可以看出互联网上的信息呈现爆炸式增长的趋势。

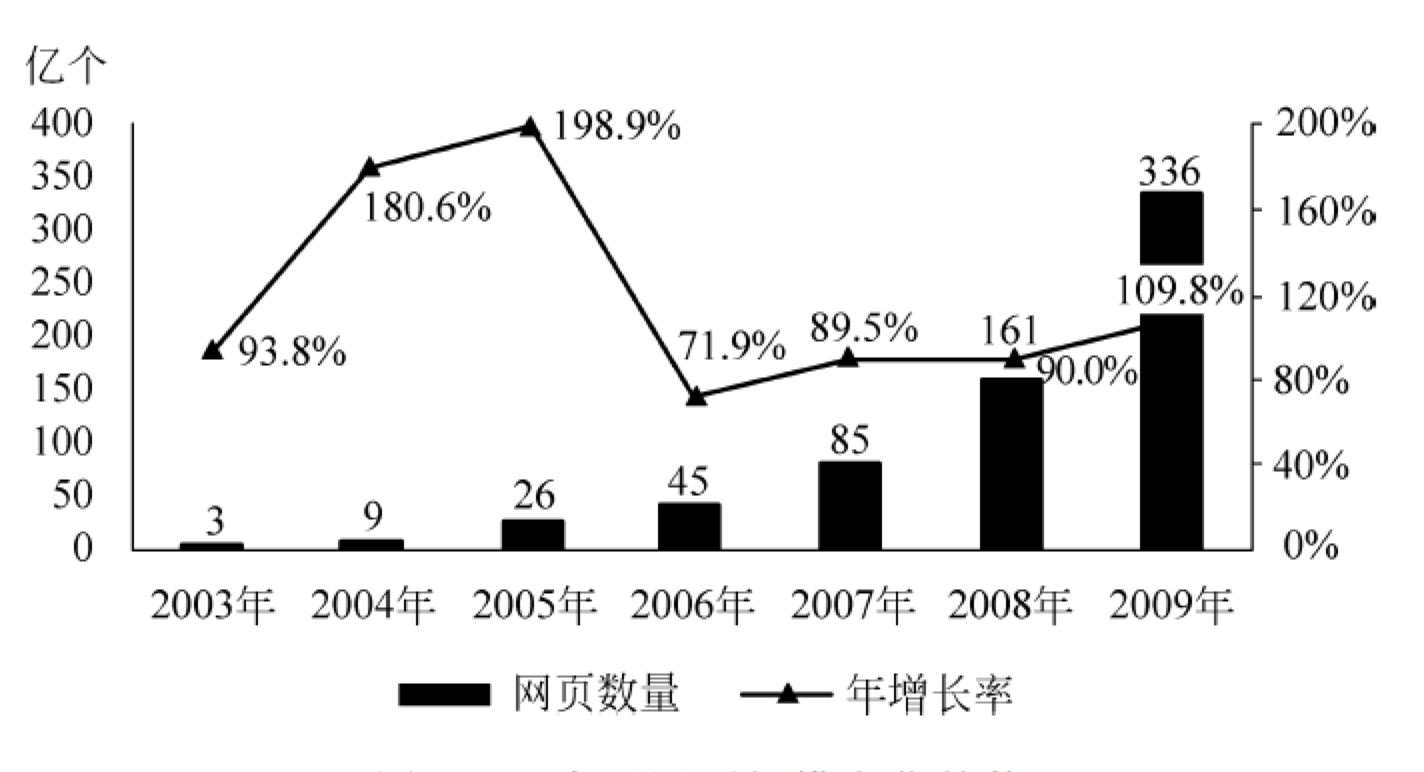


图 9.9 中国网页规模变化趋势

互联网正在深刻地影响着信息传播、获取和存储的方式,其高效性、便捷性、开放性和实时性等诸多特点使其成为信息的有效载体。人们越来越多地需要从互联网上获取信息、吸收信息和利用信息。互联网上的信息是海量的,在这些海量信息之中,必定具有部分信息是冗余的、重复的和相似的。据有关部门统计显示:互联网上相似网页占总网页数的三成,而完全相同的网页占总网页数约两成。这些相似或相同的网页有些是完全复制,有些是内容稍做修改,有些仅仅是网页格式发生变化,并不能带来实际意义,反而成为一种麻烦,带来资源的无谓消耗和信息获取效率的损失。尤其是对当前人们从互联网上获取信息的重要工具搜索引擎的影响,这些意义不大的相似网页不得不让搜索引擎付出额外的代价,即:

- 不必要的存储空间开销。如果搜索引擎能够将这些相似或相同的网页从数据库或其他存储设备中去掉,将会节省不小的存储空间以存放更多的有效网页。

- 不必要的网页搜集速度开销。如果搜索引擎能够在网页搜集过程中绕开这些相似或相同的网页,将会提高有效网页的搜集速度。
- 不必要的用户体验下降。如果搜索引擎能够返回给用户有效信息多而冗余信息少的结果,将大大提升用户体验,用户不会因为过多的接纳重复冗余信息而感到不快,提高用户信息获取效率和质量。

因此对相似网页识别技术的研究是一个值得关注的话题。根据相似网页之间的不同点,相似网页一般分成以下几种:

- 网页内容和格式上没有什么区别。
- 网页内容相同,但格式不同。
- 网页有部分重要的内容相同,并且格式相同。
- 网页有部分重要的内容相同,但是格式不同。

相似网页之间相似性的定义有三个关系上的特性:

- 自反性
- 对称性
- 非传递性

所谓的相似网页识别算法就是在给定大量的网页集中,发现相似网页的过程。由于互联网上网页数量庞大且内容结构复杂,这些都对相似网页识别算法的速度和准确度提出了很高的要求,一个又快又准的相似网页识别算法其作用的重要程度是不言而喻的。

相似网页的识别在信息检索中有着广泛的用途,特别是对于搜索引擎而言,相似网页的识别是一项重要的关键技术。在返回检索结果前,将相似网页过滤掉,可以更好地提高检索效果和用户使用体验,减少用户无谓的点击时间。从另一个角度而言,识别出来的相似网页也有一定的作用。当某网页不能访问时,可以向用户提供相似网页的地址,获取信息。

对于网络爬虫而言,可以忽略掉相似网页的外向链接,减少无用的抓取,提高网络带宽的利用率。在识别相似网页的基础之上,还可以发现镜像网站,通过去除镜像网站,可以减少搜索引擎抓取、索引和存储的开销。

通过引入专业领域知识,相似性计算还可以更好地找出专有领域的相似网页。这些知识可以大大提高主题抓取爬虫获取专有领域网页集合的效率。

相似网页的识别还可以用于信息抽取。通过聚集相似网页集合,可以从中学习和挖掘这类网页的共有模板,再自动生成模板,从无结构的网页信息中挖掘出有结构的网页信息。例如提取 IMDB(Internet Movie Database)上的电影网页,或者提取淘宝上的关于产品介绍的网页。

与文本相似算法类似,相似网页识别可以应用于网络上的抄袭复制识别等,用于保护网络上的知识产权。

1. 研究现状

目前针对相似网页识别技术的研究比较广泛,国内外相关的识别算法较多。好的相似网页识别技术能够使人们更好、更高效地使用互联网上丰富的信息资源。目前已有一些识别算法已经投入使用,取得了不错的效果,给人们带来了便利。其中比较经典的有 Broder

等提出的DSC算法、SS-DSC算法以及Charikar提出的Simhash算法等。它们都具有各自的特点和优点,但是同样存在不足之处,对这些相似网页识别算法的学习、研究、分析和改进是一个值得探讨的课题。如基于单词序列的DSC算法,该算法使用连续的单词序列表征一篇网页,考虑了单词之间的相对位置信息,但是并未融入单词的相对词义、词频以及网页结构特点等,而且基于单词的序列导致大量序列集的产生,在计算和存储方面都带来了不利影响。另外,当前诸多算法各有优点,而这些优点的有机结合将是一个值得研究的方向,追求强强联合,发挥共同的优势,避免各自的缺点。如Simhash算法中出色的文档指纹思想就可以与DSC算法中的单词序列相结合。

文本相似性计算是相似网页识别中的关键一环。文本相似性计算是自然语言处理、Web检索、文本聚类及文本分类研究中一个课题。对文本相似性计算的研究可以分为两个方向:语法相似性和语义相似性。语义相似性计算的研究涉及到自然语言理解,目前还没有比较好的处理方法,是研究上的一个待突破点。从语法出发计算文本相似性有很多方法,主要分为两大类:一类是基于字符串比较的方法,字符串可以基于不同的粒度;另一类是基于词频统计的方法,在向量空间模型上,通过降维、计算距离等度量相似性。

国际上对相似文档识别算法的研究最初是针对大型文件系统的,后来又被拓展应用于数字化图书馆和搜索引擎系统。美国Arizona大学的研究人员采用计算文档重叠程度的方法发现一个大型文件系统中的相似文本。作为斯坦福大学数字化图书馆项目的一部分,N. Shivakumar等人研制了SCAM原型系统,用于发现相似的数字化文档。后来Shivakumar和Garca-Molina在SCAM的基础上,提出了一种全文分段签名的相似网页识别算法,并将此用于Google系统。现在,世界上成功的搜索引擎一般都有一套自己的相似网页识别算法,并不断进行改善。

Broder提出了一种有效识别文本相似的方法,并将这种方法用于AltaVista搜集的互联网上的所有网页。作者首先相对于URL,定义了与位置无关的URN作为网上所有资源的标识符。然后定义了相似度和包含度,识别了所有相似网页对,并建立了所有相似网页类。并且将结果用于“失物招领”(Lost and Found)服务、网页搜索结果的过滤、广泛分布的镜像文件的更新和违法知识产权的剽窃行为的检测等应用。

后来对SCAM系统的相似文本检测算法做了进一步改进,并应用于斯坦福大学开发的Google搜索引擎。有文献提出了如何利用重复文本集合、超文本网页集来提高搜索引擎抓取器、网页文档馆和网页排名函数等应用的效果。另外有文献首先定义了相似网页集的概念,并提出了一种有效的识别相似网页集的算法,这是一种搜索最优子图算法,对于处理大量网页十分有效。用Google搜索到25 000 000个网页,约150GB的数据集进行实验。使用了不同参数的相似网页识别查重算法,得出的结论是25 000 000个网页中有36%～48%的网页重复。

王建勇利用文档的向量空间表示,为基于关键词匹配的搜索引擎提出了5种相似网页识别算法,并利用天网系统对这5种算法进行了实际评测。另外还将它们与现有的方法进行对比分析。实际评测以及与现有方法进行对比分析后,表明这些方法是非常成功的,能够以极小的时间复杂度和空间复杂度获取较高的查全率,同时保持了很高的查准率。该相似网页识别算法已经成功运用于消除天网搜索引擎系统的相似网页,同时它们也可广泛应用

于数字化图书馆的搭建。

Zhangyi 针对 Web Application 的需求，设计了一种网页预处理的框架和方法。其预处理包括三个方面：网页净化、网页去重和网页整合，并最后将网页转化为一种通用的 DocView 格式。算法首先对网页进行标签树构造，然后对标签树中的结点进行裁剪以达到净化目的。算法针对主题网页、目录型网页和图片网页的不同特征，采用不同的净化方法。该方法已经应用于天网搜索引擎的网页去重和自动分类，天网搜索引擎已经采用了这种预处理的框架。

Charika 的 Simhash 算法蕴涵了更深的理论背景，主要采用的是网页指纹识别技术，即为每一个网页生成一个指纹，网页不同，则指纹不同，但另一个更重要的属性是网页越相似，则指纹越相似，这也是该算法的核心之一。要获得一个网页的指纹信息，首先要从网页中抽取一些带权重的特征集，通过这些特征集的叠加计算得到该网页的指纹。

DSC 算法思想则是通过对网页文本信息的处理，得到 K 个连续的单词，这些连续的单词构成一个单词序列，然后网页就由一个个单词序列的集合来表征。所有的网页单词序列集合最后采用一种类似于倒排索引方式的倒排表存储，如＜单词序列，文档 ID＞以表示某个单词序列出现在网页文档中。如果两个网页非常相似，则网页中很可能会有公共的单词序列，公共的单词序列越多，则网页相似度越大，从而作为两个网页相似判定的依据。为了提高该算法的速度，Broder 等又提出了对该算法的一些改进，采用 DSC-SS 算法。这样大大减少了查找相似网页的时间和存储空间，但同时也降低了对某些内容短小文本的识别精度，因为内容短小的网页本身只含有少量的单词序列，再从这些少量的单词序列中提取超大单词序列就更少了。

Google 采用的算法是通过将网页内容的每一个 Token 分配到一个列表中去，分配法则是获得每个 Token 的哈希值，然后根据哈希值决定分配到哪一个列表，最后网页由多个这样的列表组成，最后对每个列表计算其指纹，如果两个文档中有任何一个指纹相似则判定这两个文档相似。

另外还有针对"词组级重复"的算法。有些网站为了吸引大量的搜索引擎用户访问，而将网页内大量的符合语法规则的语句或短语进行动态组合，这种组合可以由程序自动生成。如此生成的大量网页也会被搜索引擎收录到，从而诱导搜索引擎用户通过检索结果点击该网站。针对这种现象的短语级相似网页识别技术则可以避免搜索引擎爬取程序落入这种圈套。

目前相似网页识别算法可以分为以下几个主要研究方向：基于内容的相似网页识别、基于链接的相似网页识别、基于链接和内容的相似网页识别。其中基于链接的相似网页识别不够成熟，主流的是基于内容的相似网页识别。

2. 主要算法

从对相似网页处理的细分粒度看，相似网页识别算法可以分为三类：以单词为最小单位；以词组或句子为最小单位和以整篇文档为最小单位。其中，粒度越小，计算越准确，速度相对较快；粒度越大，计算越快，精确度相对较低。

从相似网页识别所利用的信息看，相似网页识别算法可以分为三类：基于文本内容的，基于文本内容和链接的以及基于文本内容、链接和 URL 的。

目前，相似网页识别算法主要是基于网页文本内容的，基于链接关系的算法尚未成熟。

1) DSC 和 DSC-SS 算法

DSC(Digital Syntactic Clustering)及其改进 DSC-SS 两种算法由 Broder 等在 1997 年提出。DSC 算法是基于单词序列的,一篇文档由 w 个单词序列组成,再根据相关的过滤策略对单词序列进行选择保留。所谓的单词序列与自然语言处理领域中定义的 N-GRAM 的概念有些类似,就是一段连续出现的具有固定长度为 N 的单词串,单词序列与 N-GRAM 这两者的不同之处在于 DSC 算法中相同的序列会被合并为同一个序列,而 N-GRAM 中并没有这种合并的过程。单词序列类似于文档的一个特征,单词序列的集合就构成了文档特征集。DSC 算法忽略了网页的标签、网页的格式及字母大小写等细节。假设 A 为一篇文档,定义 $S(A,w)$ 为文档 A 的单词序列集合,w 表示单词序列的固定长度。

定义文档 A = (a, rose, is, a, rose, is, a),设置 $w = 4$;则文档 A 的单词序列集合

$$S(A,w) = \{(a,rose,is,a),(rose,is,a,rose),(is,a,rose,is),(a,rose,is,a)\}; \quad (9.2)$$

DSC 算法对于多次出现的单词序列在 S(A,w)中只记录一次以提高效率。因此

$$S(A,w) = \{(a,rose,is,a),(rose,is,a,rose),(is,a,rose,is)\};$$

对于两篇网页文档 A 和 B 的相似度,采用 0~1 之间的数值量化表示,当该数值越接近 1 时,两篇网页文档越相似。

网页 A 与网页 B 的相似性度量 $r(A,B)$定义如下:

$$r(A,B) = \frac{|S(A,w) \cap S(B,w)|}{|S(A,w) \cup S(B,w)|} \quad (9.3)$$

$|X|$表示集合 X 中的元素个数。

网页 A 在网页 B 的包含度 $c(A,B)$定义如下:

$$c(A,B) = \frac{|S(A,w) \cap S(B,w)|}{|S(A,w)|} \quad (9.4)$$

因此,$r(A,B)$的数值在 0~1 之间,且 $r(A,A)$的值为 1。需要指出的是当 $r(A,B)=1$ 时,也只能表明 B 是 A 的任意排列中的一个,如 $A=(a,c,a,b,a)$,$B=(a,b,a,c,a)$,设 $w=2$,则 $r(A,B) = 1$;另外需要指出的一点是 $r(A,B)$不具有传递性,如 $A1$ 与 $A2$ 相似,$A2$ 与 $A3$ 相似,…,$A99$ 与 $A100$ 相似,不能够推出 $A1$ 与 $A100$ 相似,也许它们已经一点都不相同了。DSC 算法中单词序列长度即 w 的选择是一个比较重要的问题,当 w 取得过长时,文档中很小的随机变化就会带来巨大的影响,而 w 过小时,更容易将一些不相关的文档赋予更大的相似性,根据实际情况的不同,Broder 推荐 w 一般介于 3~10 之间。为了获得更快的速度,DSC 算法采用倒排索引的方法,建立排序的<序列,ID>列表和<ID,ID,共有序列数>。但是经过选择保留后的序列数量依然不少,比较次数太多、效率仍不够理想。

DSC-SS 算法是对 DSC 算法的一种改进。DSC-SS 算法在序列之上再进行一个提取大序列的过程,即把几个序列合并为一个更大的序列,这样在一定程度上减少了序列的个数,因此减低了比较的次数。如果大序列的大小选择合适,两个相似文档会有很大的概率至少拥有一个相同的大序列。DSC-SS 不仅减少了比较次数,同样还能减少对序列的存储,但是对于一些小文档,由于这些小文档本身的序列数量就不多,再进行合并,会导致准确度的下降。

2）I-Match 算法

I-Match 算法同样是对 DSC 算法的一种改进，但不同于 DSC-SS 算法，它从过滤单词序列而不是从减少比较次数这方面下手，尽量抛弃那些重复次数较多、过于普通的单词序列。这种选择抛弃策略与经典的 tf-idf 算法中的 idf 概念有些相似。将集合中的文档生成单词序列后，计算每个单词序列的出现次数，和有该单词序列出现的文档个数，依据每个单词序列的 idf 值选择抛弃，再对剩下的单词序列进行哈希运算，获得一个整型哈希值，通过对这些整型值的比较判断两篇文档是否相似。

I-Match 算法并不依赖于严格的语法分析，而是基于集合统计识别哪些单词序列将被选为进行比较的对象。

对于每个单词序列的 idf 值，定义为 $t = \log(N/n)$，其中 N 是集合文档总数，n 是包含单词序列的文档数目。通过 idf 值过滤后的单词序列能更好地描述一篇文档。

计算 idf 值的方法有两种：第一种方法就是在识别开始之前，使用训练集计算每个单词序列的 idf 值。因为当集合大小改变时，idf 值只会轻微改变，所以这是一个可以接受的方案；第二种方法是把 I-Match 算法分成两个步骤，第一步先整体计算所有单词序列的 idf 值，第二步再利用 I-Match 算法查找重复文档，虽然这种方法会增加实际的运行时间，但是其理论复杂度仍保持不变。

I-Match 算法的过程描述如下：

```
1.获取一篇文档;
2.除去文档的格式标签,将其解析为一个单词序列集;
3.依据 idf 值选择抛弃,保留下重要的单词序列;
4.将选择的相关单词序列插入到以 Unicode 升序排列存储且不重复的树结构中;
5.遍历整个单词序列树,将出现的每一个单词序列进行哈希签名.当遍历完成后,就生成一个
<DocId, 哈希签名>元组;
6.将元组插入到以哈希签名做关键词存储的数据结构中;
7.如果签名值出现冲突,那么这些冲突的文档就是相似的.
```

相比与 DSC-SS 算法，DSC-SS 算法生成 k 个大单词序列来描述一篇文档，而 I-Match 算法只产生一个，由于 k 是一个常数，所以这两种算法复杂度在理论上是相同的。

I-Match 算法对比 DSC-SS 算法的优势不在于时间效率上的提高，而是它能够弥补计算短文档时不够理想的准确率。DSC-SS 算法对于短文档不能提取出足够数量的单词序列进行相似性识别，因此即使这些短文档中出现重复，DSC-SS 算法也无法有效识别，对于很多文档相似识别技术的具体应用领域，忽略短文档会影响算法的查全率，而 I-Match 的提出对这一问题进行了有效补充。

3）基于全文分段签名匹配的相似网页识别算法

全文分段签名匹配的相似网页识别算法是把一篇网页按照一定的原则分成 m 段，如每 n 行作为一段、利用文本的自然段等，然后对每一段进行签名，于是可以利用 m 个签名来表示一篇文档。对于给定的两篇文档，如果它们的 m 个签名中存在 t 个是相同的，则认为它们是相似网页。

Shivakumar 和 Garca-Molina 提出的相似网页识别算法就是一种全文分段签名的算法，采用每 n 行作为一段的分段方法。并比较了参数 n 的选择对于识别效果的影响。例如，将全文作为一整段、每四行作为一段或每两行作为一段等。

假设网页的 MD5 值长度为 m,网页数为 n,则全文分段签名算法的空间复杂度为 $O(mn)$,时间复杂度为 $O(mmnn)$。由于互联网上存在的网页数量过于庞大,这样的时间空间复杂度不能很好地满足实际应用需求。

该算法使用对＜文档标示,段标示,指纹＞三元组排序的方法避免了对所有网页进行两两比较,使算法的复杂度有所降低,但是程度有限。

4) 基于向量空间模型的相似网页识别算法

向量空间模型(Vector Space Model,VSM)是由 Salton 等人于 20 世纪 60 年代提出的,并成功地运用于著名的 Smart 文本检索系统。

使用向量空间模型表征文本时,先抽取具有代表性的特征项,每个特征项具有各自的权重,利用向量描述文本,再依据相似性的计算得到相似性关系。向量空间模型是在信息检索领域中应用广泛、效果较好的一种文本表示模型。在传统的文本复制检测中基于词频统计的文本相似性计算就使用了基于向量空间模型的文本表示方法和相似性计算方法。

VSM 的特征提取是从自然语言形式的文本中提取最能够代表全文的特征项集合。如文本中字的集合、词语的集合、短语的集合或者句子的集合等,用这样的集合代表全文内容。对于中文,是由字组成词组、由词组组成句子,再由句子段落及篇章对文本进行层次性表示。向量空间模型通常选取词作为其特征项的基本单位,对于基于短语、句子的向量空间模型也曾进行过尝试,在文本分类中效果表现良好。

在文本自动分类、过滤等领域,特征项要求代表性强,特征项集合要求简明扼要。传统的特征项选择方法有文档扩充计量法、频次法、信息增益法、互信息法、期望交叉熵法、词频法和文本证据权法等,它们都是针对文本特征项维度过高,通过降低特征空间维度提高计算的效率和准确率。

VSM 的概念比较简单,利用向量空间中的向量运算简化对文本内容的处理,利用空间上的相似性来表达语义的相似性,直观易懂。文档被表示为文档空间的向量,通过计算向量之间的相似性度量文档间的相似性。最常用的相似性度量方式是利用夹角余弦距离。

用 D 表示文档(Document),t 表示特征项(Term),特征项是指出现在文档 D 中且能够代表该文档内容的基本语言单位,主要是由词或短语构成,文本可以用特征项集表示为 $D(t_1,t_2,\cdots,t_n)$,其中 t_k 是特征项,$1\leqslant k\leqslant n$。例如一篇文档中有 a、b、c、d 四个特征项,那么该篇文档可以表示为 $D(a,b,c,d)$。对于含有 n 个特征项的文本而言,通常会给每个特征项赋予一定的权重表示其重要程度。即 $D=D(<t_1,w_1>,<t_2,w_2>,\cdots,<t_n,w_n>)$,简记为 $D=D(w_1,w_2,\cdots,w_n)$,称为文本 D 的向量表示。其中 w_k 是特征项 t_k 的权重,$1\leqslant k\leqslant n$。在上述例子中,假设 a、b、c、d 的权重分别为 30、20、20、10,那么该文本的向量表示为 $D(30,20,20,10)$。在向量空间模型中,两个文本 $D1$ 和 $D2$ 之间的相似度 $\mathrm{Sim}(D1,D2)$ 常用向量夹角余弦值表示,计算公式为:

$$\mathrm{Sim}(D1,D2)=\cos\theta=\frac{\sum_{k=1}^{n}W_{1k}\times W_{2k}}{\sqrt{\left(\sum_{k=1}^{n}W_{1k}^2\right)\times\left(\sum_{k=1}^{n}W_{2k}^2\right)}} \tag{9.5}$$

其中，W_{1k}、W_{2k} 分别表示文本 $D1$ 和 $D2$ 第 k 个特征项的权值，$1 \leqslant k \leqslant N$。

数学上，n 维向量是 $V\{v1, v2, v3, \cdots, vn\}$

它的模：

$$|V| = sqrt(v1 * v1 + v2 * v2 + \cdots + vn * vn) \tag{9.6}$$

两个向量的点积

$$m * n = n1 * m1 + n2 * m2 + \cdots + nn * mn \tag{9.7}$$

$$\text{相似度} = (m * n)/(|m| * |n|)$$

物理意义就是两向量的空间夹角余弦值。

向量空间模型可以实现文档的自动分类和对查询结果的相似性排序，能够有效地提高检索效率，缺点是相似性计算量大，当有新的文档加入时，则必须重新计算词的权重。

5）基于特征串的相似网页识别算法

哈工大于 2003 年提出一种基于特征索引的相似网页识别算法，其中提取网页特征码是比较重要的环节，网页特征码必须能够区分出完全相同的网页和不同或相似网页，普通的关键词技术是很难做到这一点的，因此特征码最好取网页中一个长度固定的词串。但由于相同的网页内容中的版权和导航信息有可能并不同，因为这些信息的干扰，很难从网页的开始或者中间的某个固定的位置抽取特征码。通过网页分析发现标点符号尤其是句号很少在导航类信息中出现，另外导航信息一般以超链接的形式出现在网页中。利用这两个特点，在提取特征码时可以尽量将导航信息等去掉。将句号作为一个提取的分隔符，在句号两边提取固定长度的两个词串从而构成网页的特征码。在句号两边提取的一个好处就是句号两边的字很难构成一个词，更有利于保持特征码的唯一性。

特征码提取中固定长度的选择对提取效果影响较大，长度过长不利于存储和计算，带来的开销较大，长度太短将会降低特征码的区分能力。一般地，可以将固定长度设置为 10，即在句号两边各选择五个汉字作为特征码，对于不足 5 个汉字的情况，以特定的字符进行补位。按照信息论中的多元文法的定义这个特征码相当于一个 10 阶的文法，按照 6763 个汉字计算，这个特征码重复的概率大约为 6763 的十次方分之一，因此保证了这个特征码在不同网页中几乎是不可能重复的。从另一个方面来说，如果把长度为 L 的特征码看成一个二进制，即把一个网页映射到一些二进制数字构成的空间中，而这个空间的大小为 6763 的十次幂，在这样一个大的空间里，不同网页出现相同特征码的可能性极小。通过这种映射可把一个复杂的计算网页相似度来判断网页是否相同的问题变换为利用二进制数的比较来判断网页内容是否相同的问题，从而克服了因为相似度域值不合适而产生的误判。

在特征码提取完成后，使用平衡多路查找树有效地组织存储这些特征码。生成平衡多路查找树的过程就是对特征码的插入过程。插入特征码键值时，首先查找待插入的结点，然后插入对应的位置，如果待插入的特征码键值与树中已经存在的特征码的键值相同，说明这个键值特征码所代表的网页与结点中键值所代表的网页重复，此时需要记录下这对网页。如果平衡多路查找树中未发生键值冲突，就将待插入键值特征码插入到平衡多路查找树中。这样平衡多路查找树中的特征码键值都不存在重复冲突的情况，从而不会重复索引相同的网页。所有特征码都被插入完毕后，即完成对相似网页的记录操作。

该方法在小规模数据集上的查准率和效率都比较理想,并且简单、高效,从工程角度上看在大规模相似网页识别上应该有广阔的应用前景。

6) Simhash 相似网页识别算法

Simhash 算法是由 Charikar 等人提出的比较经典的相似文本识别算法。Simhash 算法基于文档特征及特征权重,首先维护一个初始值为 0 的 f 维向量,然后遍历特征集中的每个特征,使用 Rabin 哈希函数对每一个特征生成一个 f 位的哈希值,遍历 f 位哈希值中的每一位,如果哈希值中第 k 位的值为 0,则将被初始化后的 f 维向量中的第 k 维减去该特征对应的权重值,若哈希值第 k 位为 1,则加上该特征对应的权重值。当遍历完所有的特征后,初始各位为 0 的 f 维向量中各维的值已经发生了变化,其中有正有负,对于负值的位以 0 替换,正值的位以 1 替换,这样将形成一个 f 位的二进制串,该串就定义为该文档的特征指纹,Simhash 算法的描述如下:

```
初始化 f 维向量为 0; 同时初始化 f 位的二进制数 s 各位为 0;
遍历特征项集,用哈希函数对每一个特征生成一个 f 位的数字签名 b.
对 i = 1 到 f:
    ◆ 若 b 的第 i 位为 1,则 V 的第 i 个元素加上该特征的权重值;
    ◆ 否则,V 的第 i 个元素减去该特征的权重值.
如果 V 的第 i 个元素大于 0,则 s 的第 i 位为 1,否则为 0;
输出 s 作为特征指纹.
```

算法过程如图 9.10 所示。

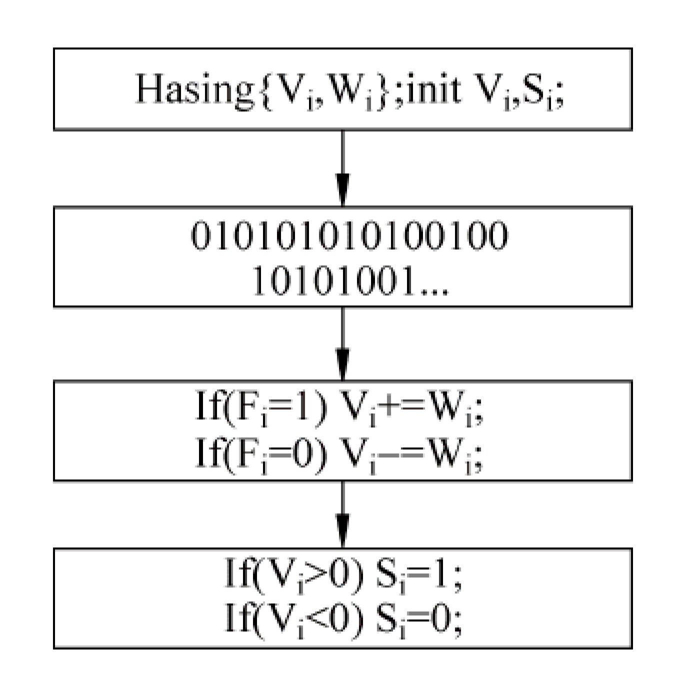


图 9.10 Simhash 算法过程

Simhash 指纹具有两个非常重要的特点:一是一个文档由一个指纹表征,该指纹为文档特征集的哈希值;二是相似文档有着相似的指纹。

Simhash 算法将高维向量用较低维度的签名表示,衡量两篇文档内容的相似度,通过计算指纹间的海明距离,海明距离可以用来衡量原始向量间的夹角。

假设文档 A 的指纹为 $f_p(A)$,文档 B 的指纹为 $f_p(B)$,则 A 与 B 的相似度用 $f_p(A)$ 与 $f_p(B)$ 之间的海明距离表示,即 $f_p(A)$ 与 $f_p(B)$ 二进制位中不相同的位的个数,如果海明距离越小,文档 A 与 B 的相似性越大。

从空间复杂度看,每一个网页文档通过 Simhash 计算后都由一个 64 位的二进制哈希值代替,假设处理的网页文档数位 N,则总共的空间消耗为 8NB,这个存储空间是相对较低的。对于计算每篇网页文档的 Simhash 数值,假设文档词数为 n,则时间复杂度为 $O(n)$。判断两个文档是否相似,则看它们哈希值的海明距离是否小于等于 3。对于一篇给定文档 D,查找其相似文档,就是找出哈希值相差 3 位以内的文档,最简单的方法就是对于一个给定的文档的哈希值 S,取 $C=41\ 644$ 个与 S 相差 3 位或以内的哈希值,若这些哈希值有对应的文档,则为 D 的相似文档。

7) 基于 Web 链接结构的相似网页识别算法

Web 网页与文本的区别之一就是拥有自身的格式特征,除此之外更重要的一点是具有文本所没有的链接结构。因此,如何挖掘 Web 网页之间的链接信息作为网页相似性的度量,也是近年来的研究热点之一。

与基于文本的网页相似性识别不同,基于链接的网页相似性识别考虑的不是网页内容特征,而是网页之间的相邻结点的结构信息。比如 $I(a)$表示指向网页 a 的相邻结点集合,$O(a)$表示从 a 指出的相邻结点集合。$\text{Sim}(a,b)$表示网页 a 与网页 b 的相似性。

如图 9.11 所示,采用共同邻居的方法,则

$$\text{Sim}(a,b) = |O(a) \cap O(b)| = |(c,d)| = 2$$

如图 9.12 所示,采用共同引用的方法,则

$$\text{Sim}(a,b) = |I(a) \cap I(b)| = |(c,d)| = 2$$

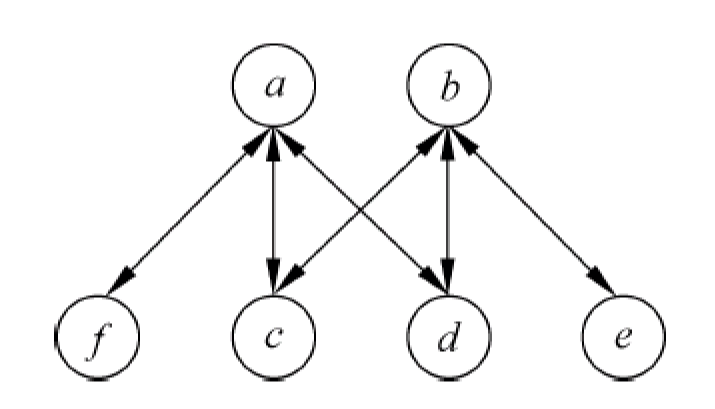


图 9.11 基于链接的相似性度量

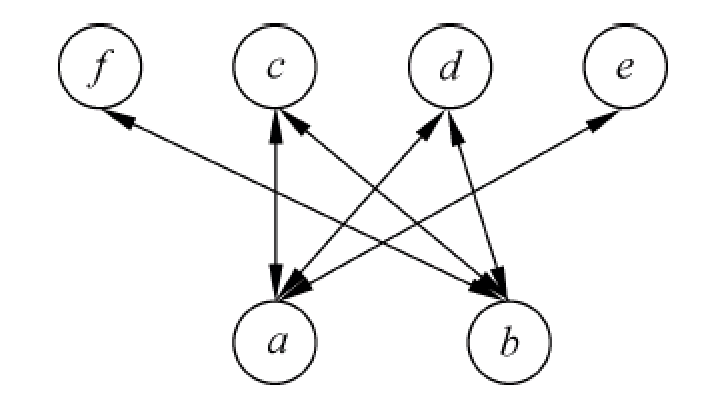


图 9.12 基于链接的相似性度量

这两种方法考虑的只是简单的一层邻居关系,而没有利用整个网络图结构信息,因此不能很好地作为网页相似性度量方法,而且对于 Web 而言,并不是所有的链接都是同等重要的。

在图链接关系理论的基础上,基于这样的直觉"如果两个结点被相似的结点引用,那么它们也是相似的",斯坦福大学的 Glen Jew 和 Jennifer Widen 的 SimRank 算法成为基于链接的网页相似性度量的著名方法。

SimRank 算法采用了递归的定义,即:

对于相同的网页,$\text{Sim}(u,u) = 1$;

对于不同的网页,如果$|I(u)||I(v)| = 0$, $\text{Sim}(u,v) = 0$,否则

$$\text{Sim}(u,v) = C\frac{\sum_{a\in I(u)}\sum_{b\in I(v)}\text{Sim}(a,b)}{|I(u)||I(v)|} \tag{9.8}$$

这里 C 是一个介于 0 和 1 之间的常数,迭代计算从初始条件 $\text{Sim}(u,u)=1$,$\text{Sim}(u,v)=0$ 开始,u 不等于 v。

基于 SimRank 和 PageRank,香港中文大学的 Zhangjiang Lin 等人提出了一种新的基于链接的网页相似性识别算法 PageSim。

PageSim 的提出是为了解决 SimRank 不能计算任意两个网页之间相似性识别的问题,并且采用 PageRank 分值传播的策略,对于不同的链接会不同对待。PageSim 模型更加适合于网页相似性的计算,它是高效的、可扩展的、稳定的并且鲁棒的计算模型。PageSim 的思想是使用 PageRank 值度量一个网页权威性,之后每一个网页将其自身的相似度值传播给它的相邻结点,每一个网页也同时将其他的网页相似性值传播给邻居。当相似度值传播结束后,每一个网页得到的就是一个相似度值数组。

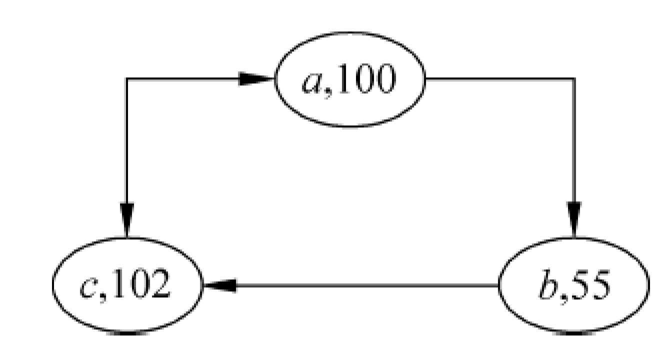


图 9.13 PageSim 计算

举个简单的例子,如图 9.13 所示,开始时分别为 a、b、c 三个结点赋 PageRank 值,$\text{PR}(a)=100$,$\text{PR}(b)=55$,$\text{PR}(c)=102$。计算时,每个网页把它的相似度分值的 80%平均地传播

给其邻居。假定第一次传播从 a 点开始,a 将其相似度值 80 平均传给 b 和 c,这样 b 和 c 都为 40,接下来 b 再把它的 80%即 32 分传给 c,c 点只有链接到 a,因此这轮传播结束,三点的相似度值分别为 100、40 和 72。同样的从 b 和 c 出发开始传播,最后每一个网页都会得到一个相似度分值向量 SV,按照图计算,最后得到:

$$\mathrm{SV}(a) = (100,35,82)$$
$$\mathrm{SV}(b) = (40,55,33)$$
$$\mathrm{SV}(c) = (72,44,102)$$

任意两个结点的 PageSim 分值取两个结点 SV 向量每一维最小值之和,例如 a 和 b 之间的 PageSim 值为 $\mathrm{PS}(a,b) = 40+35+33 = 108$。

如果两个网页之间有更高的 PageSim 相似度分值,则它们会更相似。

8) SIF

1993 年,Arizona 大学提出了 SIF 工具,其主要应用是在大型文件系统中查找相似文档,包括程序重用、信息收集、文件管理、复制识别、数据压缩和文件同步。它提出了相似指纹的概念,利用相似指纹表示文本和计算文本相似性。相似指纹完成了对文档内容的一种压缩,在通常情况下,使得相似文本的指纹差别小,而不相似文本的指纹差别大。

SIF 工具进行相似度计算的主要过程如下:先提取一定的字符串作为"指纹",通过哈希运算将不同的字符串映射为不同的数值,再比较不同文档的数值集合,如果代表两篇文档的数值组合中相同的数字超过一定的阈值,则判断两篇文档相似。这样,"近似指纹"计算将字符串的匹配问题转换成了数值比较问题。

核心的处理过程是以 50 个字节(字符)为长度,将文件划分为重叠的一个个子串,然后计算每一个子串的指纹值。比如把文本标识为 $t_1 t_2 \cdots t_n$,那么第一个 50 字节的字串的指纹值为:

$$F_1 = (t_1 \times P^{49} + t_2 \times P^{48} + \cdots + t_{50}) \bmod M \tag{9.9}$$

这里 P 和 M 为常数,根据霍纳法则,优化计算多项式,即:

$$F_2 = (p \times ((\cdots(p \times (p \times t_1 + t_2) + t_3)\cdots)) + t_{50}) \bmod M \tag{9.10}$$

而在计算 E 时,只需要加上最后一个参数并且减去第一项,即:

$$F_2 = (p \times F_1 + t_{51} - t_1 \times P^{49}) \bmod M \tag{9.11}$$

总之,计算的指纹数和文本的字符数成正比,而与指纹的大小无关。有很多方法可以决定挑选哪些指纹,最简单的方式是选择以 k 个 0 结尾的指纹值。近似地,2^k 个字符中会选出一个指纹。将 P 设为一个素数,M 设为 2^{30},k 设为 8。因为被选到的指纹值最后 8 位都是 0,所以可以将指纹值右移 8 位存储以节省空间。如果系统中的文件数量非常多,那么可以通过使用更大的指纹值(比如 2^{31} 或 2^{32})来减少指纹冲突的几率。

虽然通过上述方法选取了较为稀疏的指纹值,但这个问题并不重要,因为这些选取的指纹已经足以标识文本。尤其在不相关的两个文件中出现相同的 50 字节子串,这种可能性是相当小的。通过这种近似选择还可以过滤掉噪声信息。如果两个文件拥有足够数目的相同指纹,那么在一定程度上已经足以证明两个文件是相似的。

最后列举了两种检索方法:从多个中找与一个相似的(All_One)和从多个中找与多个相似的(All_All)。通过对指纹值建立倒排索引和排序等技术,可以加快检索的速度。

"近似指纹"在以后的文本相似度检测中也得到广泛应用,如 1996 年提出的 KOALA

系统和 Broder 提出的 Shingle 系统，都是基于近似指纹的字符串匹配的算法进行相似度检测的。

这些文本相似度算法都是基于语法级相似(syntactic similarity)，而没有考虑到语义级相似(semantic similarity)。换言之，对于相同的信息使用不同的词语描述，从语法级别来看，它们仍是不相似的。

9) SCAM

Garcia、Molina 和 Shivakumar 等人又提出了 SCAM(Stanford Copy Analysis Method)原型改进 COPS 系统，用于发现知识产权冲突。SCAM 借鉴了信息检索技术中的向量空间模型，使用基于词频统计的方法度量文本相似性。后来 Garcia、Molina 和 Shivakumar 等人还在 SCAM 的基础上提出了 dSCAM 模型，把检测范围从单个注册数据库扩展到分布式数据库以及在 Web 上探测文本复制的方法。

SCAM 方法受到了信息检索技术的启示。它首先统计文档中各个单词出现的次数，然后按照信息检索中常用的倒排索引存储法(inverted index storage)存储文档与词频信息。最后，SCAM 参照向量空间模型提出了相对频率模型(Relative Frequency Model，RFM)，用以度量文档相似性。向量空间模型一般采用点积或余弦公式度量相似性。而相对频率模型其实是对余弦公式进行改进，试图提高文件复制检测精度。令 D 表示候选文档，Q 表示待检测(或者查询)文档，$F(D)$表示文档 D 的词频向量，$F(Q)$表示文档 Q 的词频向量，a 表示各词的权重向量，则 VSM 利用余弦公式计算的相似度 $S_v(D,Q)$为：

$$S_v(D,Q) = \frac{\sum_{i=1}^{N} a_i^2 F_i(D) F_i(Q)}{\sqrt{\sum_{i=1}^{N} a_i^2 F_i^2(D) \sum_{i=1}^{N} a_i^2 F_i^2(Q)}} \tag{9.12}$$

显然，$S(D,Q)=S(Q,D)$。RFM 首先定义了一个靠近集(closeness set)$c(D,Q)$，用于选取文档 D 和 Q 中出现频度相近的单词。即 $c(D,Q)$包含的单词是 D 和 Q 中都有的单词，并且满足如下公式：

$$S_v(D,Q) = \frac{\sum_{i=1}^{N} a_i^2 F_i(D) F_i(Q)}{\sqrt{\sum_{i=1}^{N} a_i^2 F_i^2(D) \sum_{i=1}^{N} a_i^2 F_i^2(Q)}} \tag{9.13}$$

其中，$s=(2,+\infty)$是一个用户可调的参数。然后再计算 D 对 Q 的子集度或包含度 Subset(D,Q)，

$$S_v(D,Q) = \frac{\sum_{i=1}^{N} a_i^2 F_i(D) F_i(Q)}{\sqrt{\sum_{i=1}^{N} a_i^2 F_i^2(D) \sum_{i=1}^{N} a_i^2 F_i^2(Q)}} \tag{9.14}$$

显然，D 对 Q 的包含度与 Q 对 D 的包含度不一样，即 Subset$(D,Q)\neq$Subset(Q,D)。所以，RFM 最终的相似度 S(D,Q) =max{Subset(D,Q),Subset(Q,D)}。如果 $S(D,Q)>1$ 则令 $S(D,Q)=1$。采用 RFM 方法可以更好地检测子集包含式复制，并且 S 越大，表示对两篇文档中共有单词的容忍度越大，但是无关文档的匹配机会也会越大，即正误差(false

positives)越大；S 越小，正误差越小，但是检测小程度重合文档的能力也越小。SCAM 并未确定一个普适的 s 值，但是认为 $s=2.5$ 对于网络新闻比较合适。

9.4 Web 抽取和表示

9.4.1 Web 抽取

Web 抽取是将非结构化 Web 按照一定的需求抽取成结构化数据。例如互联网是一个巨大的资源，同一主题的信息通常分散存放在不同网站，表现的形式也各不相同，若能将这些信息收集在一起，采用结构化形式储存是十分有意义的。

实现 Web 抽取的方法主要包括两种：一种比较简单的是采用模板，另一种是不依赖网页的网页库结构化抽取。

模板是事先对特定的 Web 配置模板，设置需要的信息，可以针对有限多个网站的信息进行精确采集。其优势是简单、精确、技术难度低、便于快速部署，缺点是需要针对每个信息源的网站单独设置模板，在信息来源多样的情况下维护量巨大，所以适合少量信息源的情况，不是搜索引擎级的应用，很难满足用户查全率的要求；网页库结构化抽取方法是采用 Web 结构分析与智能分析转换的方法，自动抽取结构化的数据。其优势是可对任意的正常的 Web 进行抽取，不用对具体网站事先生成模板，对每个 Web 自动实时地生成抽取规则，完全不需要人工干预，因此抽取准确率高，不是机械的匹配，准确率达到 98%以上。由于采用页面的智能分析技术，去除了垃圾块，降低分析的压力，使处理速度大大提高，通用性较好，易于维护，一般的非专业人员经过简单培训即可。缺点是技术难度大，前期研发成本高，周期长，适合网页库级别结构化数据采集和搜索的高端应用。

9.4.2 Web 表示

Web 挖掘首要解决 Web 异构数据的集成和查询问题，因此需要一个模型清晰地描述 Web。针对 Web 半结构化的特点，寻找一个半结构化的模型是解决问题的关键。除了定义一个半结构化数据模型外，还需要一种半结构化抽取模型，即自动地从现有数据抽取半结构化 Web。Web 挖掘以半结构化抽取模型和半结构化数据模型为前提。

基于 XML 的新一代 WWW 直接面向 Web 数据，不仅可以很好地兼容原有的 Web 应用，而且可以更好地实现 Web 数据的共享和交换。XML 可看作一种半结构化的数据模型，可以很容易地将 XML 的文档描述与关系数据库的属性一一对应起来，实施精确地查询与抽取。

Web 表示常用的模型是布尔模型、概率模型和向量空间模型。其中，向量空间模型在一般的 Web 挖掘中最常用。

1. 布尔模型

采用布尔表达式对 Web 进行标识。布尔模型在传统的信息检索中有着广泛的应用，通过与用户给出的检索式进行逻辑比较检索文档，本质上是一种基于关键词的匹配。在标准

的布尔模型中，Web 文档表示为 $D(W_{i1},W_{i2},\cdots,W_{in})$，其中 n 为特征项的个数，W_{ik} 的值为 0 或 1，分别表示特征项 k 在文档 i 中是否出现。

2. 概率模型

概率模型考虑词与词的相关性，把 Web 文档集合中的对象分为相关的和无关的。基于概率，通过对词赋予某一概率值表示其在相关文档和无关文档出现的概率，然后计算文档之间的相关概率，系统依据词概率做出决策。

概率模型有多种形式，常见的是第二概率模型，其基本思想是词的概率一般是重复若干词相关性计算，每重复一次，就由用户对检出文档进行人工判断。然后利用这种反馈信息，根据词在相关文档集合和无关文档集合的分布计算其相关概率。在该模型中，词的权值计算公式为：

$$\log\frac{p(1-p)}{p'(1-p')} \tag{9.15}$$

其中，p 和 p' 分别表示某词在相关文档集合和无关文档集合中出现的概率。某一文档的权值（决定其排序的位置）则是它所含标引词的权值之和，因此文档与用户查询相关概率可定义为：

$$S(Q,D)=\sum_{i=1}^{n}\frac{p_i(1-p_i)}{p'_i(1-p'_i)} \tag{9.16}$$

概率模型的主要优点是：

- 采用严格的数学理论为依据，实现匹配。
- 采用相关反馈原理。

主要缺点是：

- 增加存储和计算的开销。
- 参数估计难度较大。

3. 向量空间模型

向量空间模型（Vector Space Model，SVM）是 20 世纪 60 年代末由 Gerard Salton 等提出的，是一个统计模型。该模型以特征项作为 Web 表示的基本单位。在向量空间模型中，Web 的内容被形式化为多维空间中的一个点，表示为向量的形式，正是因为把文档以向量的形式定映射到实数域，极大地提高了文档的可计算性和可操作性。

定义 9.1：文档（Document）

泛指一般的文献或文献中的片段（段落、句子组或句子），一般指一篇文章。

定义 9.2：项（Term）

当文档的内容被简单地看成是其含有的基本语言单位（字、词、词组或短语等）所组成的集合时，这些基本的语言单位统称为项，即文档 D 可以用项集（Term List）表示为 $D(T_1,T_2,\cdots,T_n)$，其中 T_k 是项，且 $1\leqslant k\leqslant n$。

定义 9.3：项的权重（Term Weight）

对于含有 n 个项的文档 $D(T_1,T_2,\cdots,T_n)$，项 T_k 常被赋予一定的权重 W_k，表示其在文档中的重要程度，即 $D(T_1,W_1;T_2,W_2;\cdots;T_n,W_n)$。有时在特征词条确定时，常简记为 $D=D(W_1,W_2,\cdots,W_n)$。

定义 9.4：向量空间模型

给定一个文档 $D=D(T_1,W_1;T_2,W_2;\cdots;T_n,W_n)$，由于项 T_k 在文档中既可以重复出现也存在先后次序的关系，分析起来有一定难度。为了简化，可以暂不考虑 T_k 在文档中的先后顺序并要求 T_k 互异(即没有重复)。这时可以把 $T_1,T_2,\cdots,T_n$ 看成一个 n 维的坐标系，而 $W_1,W_2,\cdots,W_n$ 为相应的坐标值，因而 $D=D(W_1,W_2,\cdots,W_n)$ 看成是 n 维空间中一个向量，称 $D(W_1,W_2,\cdots,W_n)$ 为文档 D 的向量表示。

定义 9.5：文档特征向量(Feature Vector)

VSM 中每一个文档都可以用一个向量表示，向量由项(词条)及其权重组成。该向量称为文档的特征向量，特征向量是文档的一个特征表示，在某种意义上可以完全代表文档的特性。

在 VSM 中，每一文档被映射成多维向量空间中的一个点，从而将文档的表示和匹配问题转化为向量空间向量的表示和匹配问题。

VSM 模型的不足之处是将 Web 表示成向量，作为向量空间的一个点，然后通过计算向量间的距离进行分类，一般不考虑向量中各个特征间的关系。这使得距离的计算不够准确，从而导致分类精度不高。

9.5 Web 特征提取

Web 挖掘中，通常以特征项组成的向量表示 Web 文档。但如果不加以筛选，特征项的数量可能会有成千上万，以 Web 挖掘中的 Web 分类为例，对其至少会造成两方面的不利影响。

(1) 许多 Web 挖掘不能处理如此高维的特征向量。例如 Bayes 分类器，即使是利用了独立性假设的 Naive Bayes 方法(这个假设在实际中通常是不正确的)面临这样的特征向量，其计算量也非常巨大。

(2) 特征向量中的有些词对于 Web 分类的作用非常小，可以说绝大多数的词对于 Web 分类是没有什么作用的，特别是考虑到训练文档的个数非常有限。相反，过多的特征项通常会带来负面的影响，这是因为特征项越多，利用有限的训练文档估算特征项的概率分布就越不准确。

鉴于上述原因，在 Web 分类之前先进行特征提取，不仅能减小 Web 分类的复杂度，而且对于提高最终的分类精度也会有所帮助。

在 Web 特征提取算法中，通常是构造一个评价函数，对特征集中的每个特征进行独立的评估，这样每个特征都获得一个评估分(又称为权值)，然后对所有的特征按照其权值大小排序，选取预定数目的最佳特征作为特征子集，即作为文档的主题词提出。

特征提取主要用于排除那些被认为无关或关联性不大的特征(如术语)，提出可以代表 Web 的特征，为后续的 Web 挖掘奠定基础。目前，特征提取方法很多，如词频统计的方法和 TFIDF 等。

1. 词频统计

词频统计算法非常简单，即合并重复出现的特征项，计算文档 $D(T_1,W_1;T_2,W_2;\cdots;$

T_n, W_n)中对应项 T_i 的权重 W_i,W_i 为项 T_i 在文档 D 中出现的次数,并通过设定适当的权重阈值提取文档特征。

2. TFIDF

TFIDF 是 Salton 和 McGill 在 1983 年针对向量空间信息检索范例(Vector space Information Retrieval Paradigm)提出的特征表示方法,其中,TF(Term Frequency)为特征项的文档内频度,即特征项 T_i 在文档 d 中出现的次数,记为 $\mathrm{tf}(t_i, d)$; DF(Document Frequency)为特征项的文档频度,即在文档集合中出现 T_i 的文档数,记为 $\mathrm{df}(t_i)$; IDF(Inverse Document Frequency)为特征项的反文档频度,即

$$\mathrm{idf}(t_i) = \log(n/\mathrm{df}(t_i)) \tag{9.17}$$

其中,n 表示训练样本的总数。

一般地,TFIDF 中文档的向量表示 $D(W_1, W_2, \cdots, W_n)$中对应项 T_i 的权重 W_i 定义为:

$$W_i = \mathrm{tf}(t_i, d) \times \mathrm{idf}(t_i) \tag{9.18}$$

使用 TFIDF 的明显优势是随着特征项 T_i 在文档 d 中出现次数的增加,$\mathrm{tf}(t_i, d)$增大,这样特征项 T_i 的权重也随之增大,这与通常的理解相一致; 如果特征项 T_i 在训练样本集合的许多文档中出现,则 $\mathrm{idf}(t_i)$将减小,这样特征项 T_i 的权重也随之减小,其对分辨文档类别的作用也将减小,这是因为如果特征项 T_i 的文档频度很高,则它分辨这些文档的能力相对减弱,所以使其权重减小,而那些更能分辨 Web 类别的特征项权重将增大。

采用 TFIDF 算法在一定程度上减少了常用词对文档特征抽取的影响,突出重要的特征项,同时又考虑了整个文档集合中文档之间的关系,因此提取的特征具有较高的代表性。

3. 互信息

互信息是统计模型中衡量两个随机变量 X 和 Y 之间关联程度的常用参数。互信息越大,两个特征项之间的共现性越大,同样,词和类别的互信息越大,说明词和类别的关系就越密切。可根据词和类别的互信息(Mutual Information,MI)进行特征提取。

利用互信息提取特征的过程如下:

(1) 初始情况下,该特征项集合包含所有该类中出现的词。

(2) 对于每个词,计算词和类别的互信息,即:

$$M(W, C_j) = \log_2\left(\frac{P(W \mid C_j)}{P(W)}\right) = \log_2 \frac{p(W, C_j)}{p(W) \times p(c_j)} \tag{9.19}$$

其中 $P(W, C_j)$是训练样本中特征项 W 出现在类别 C_j 的频率,$P(W)$是训练样本中特征项 W 出现的频率。

(3) 对于该类中所有的词,依据计算出的互信息排序。

(4) 抽取一定数量的词作为特征项,具体需要选取多少特征项,目前没有很好的确定方法,一般采用先定初始值,然后根据实验测试和统计结果确定最佳值。

(5) 将每类中所有的训练样本,根据抽取的特征项,进行向量维数压缩和精简。

9.6 Web 聚类

从国内外发展来看,Web 分类已取得了显著的成效,尤其是美国、德国及英国等对此领域的探讨使文本分类在理论和实践上都取得很大的进步。相对而言,Web 聚类处于起步和发展阶段。目前,Web 聚类主要应用于模式识别、空间数据分析(在地理信息系统中,通过聚类发现特征空间来建立主题索引;在空间数据挖掘中,检测并解释空间中的簇)、图像处理、经济学(尤其是市场研究方面)以及互联网中的文档分类和分析 Web 日志数据以发现相似的访问模式等。

Web 聚类是把一堆 Web 文档自动划分成不同的类别,任一类别内的 Web 文档与同类别内的其他文档的相似度要大于该文档与其他类别文档之间的相似度。

一般地,Web 聚类包括四个步骤:

(1) 模式表示,包括特征抽取以及把 Web 文档表示成可计算的形式;

(2) 根据领域知识定义模式之间的距离度量公式;

(3) 聚类或者分组;

(4) 评价结果。

目前,Web 聚类面临的主要挑战是:

- 一个 Web 文档可能包含多个主题,允许属于不同主题的文档归入多个不同的簇。
- 高维诅咒问题,由于文档特征项维度众多而造成处理效率严重降低。
- 海量文档的处理效率。
- 聚类效果评价。

Web 聚类在智能信息检索、话题检测与跟踪和自动文摘等领域都有非常广泛的应用。特别是在大规模情报分析、企业竞争情报、敏感社区发现和股情分析等方面具有很大优势,成为人们广泛研究和使用的 Web 挖掘工具之一。

Web 聚类的目标是使类内的距离尽可能的小,类间的距离尽可能的大,即相似的文档(距离小)尽可能聚在一起,不相似(距离大)的文档分成不同的类。

相似度的计算方法有很多种。在采用向量空间表示 Web 的模型中,可以通过计算两个向量之间的相似度求得文档与类别之间的相似度,经常采用的是欧式距离,即:

$$D(S,T)=\{\sum_{i}(S_i-T_i)^2\}^{1/2} \tag{9.20}$$

按照 Web 文档表示方法的不同,可将现有的 Web 聚类分为基于词(Word-based)的、基于知识(Knowledge-based)的和基于信息(Information-based)的三类。

(1) 基于词(Word-based)

理论上,文本自动处理是以概念为基本处理单元的,而词是概念的基本组成部分,是不可再分的基本表意单元,是信息的基本载体。因此用词代表文本显然是可行的。这种方法需要较好的切分技术对文本进行切分,在此基础上这种方法的关键是合理选取那些可以代表文本主题内容的词汇,并据此对文本进行自动类别判定。

(2) 基于知识(Knowledge-based)

这种基于知识的文本聚类方法主要依赖于一个明确的知识库。知识的表示方法主要包

括产生式、语义网络、框架、谓词、面向对象、粗糙集、神经网络、基于语言场和基于知识本体表示法等。基于知识的聚类的显著特点是需要手工建造知识库，且建造的知识库领域性极强，可移植性较差。有研究表明，在一定领域内，基于知识的自动聚类能够快速准确地进行文本归类。

（3）基于信息（Information-based）

基于信息的聚类是一种介于基于词和基于知识的聚类之间的方法。该方法对上下文敏感，是一种有选择的概念抽取技术。用于文本聚类中，只抽取那些对文本分类有用的信息，即抽取短语及短语周围的文本和潜在的语义信息进行文本类别的确定。需要指出的是，这种方法可以用来处理没有关键词或关键短语的文章，并且避免了基于词的文本自动归类在处理一词多义、一义多词、短语、局部文本以至全文文本的局限性。

实现 Web 聚类的方法不少，主要包括平面划分聚类、层次聚类、基于网格的方法、基于密度的方法和基于模型的方法等。下面简要介绍层次聚类法和平面划分法。

1. 层次聚类法

层次聚类法是建立在给定数据集合的一个层次性的分解，根据层次分解的形成过程，这类方法可分为分裂（自顶向下）的或合并（自底向上）的。为了弥补合并或分裂的严格性，层次聚类方法的聚类质量可以通过分析每个层次划分中的对象链接，或集成其他的聚类方法加以改进。

对于给定的 Web 文档集 $D=\{d_1,\cdots,d_n\}$，层次聚类的具体过程为：

（1）将文档集 $D=\{d_1,\cdots,d_n\}$ 中的每一个文档 d_i 看作是一个具有单个成员的类 $C_i=\{d_i\}$，这些类构成 D 的一个聚类 $C=\{c_1,\cdots,c_n\}$；

（2）计算 C 中每对类 (c_i,c_j) 之间的相似度 $\mathrm{sim}(c_i,c_j)$；

（3）选取具有最大相似度的类对 $\arg\max \mathrm{sim}(c_i,c_j)$，并将 c_i 和 c_j 合并为一个新的类 $c_k=c_i\cup c_j$，从而构成 D 的一个新的类 $C=\{c_1,\cdots,c_{n-1}\}$；

（4）重复上述步骤，直到 C 中只剩下一个类为止。

该过程构造出一棵生成树，其中包含了类的层次信息以及所有类内和类间的相似度。层次聚类是最常用的 Web 聚类方法，它能够生成层次化的嵌套类，而且准确度高。但是，在每次合并时，需要全局地比较所有类之间的相似度，并选择出最佳的两个类，因此运算速度较慢，不适合大量文档的集合。

2. 平面划分法

平面划分聚类法与层次聚类法的区别在于，它将文档集合水平地分割为若干类，而不是生成层次化的嵌套类。它首先得到初始 k 个划分的集合，参数 k 是划分簇的数目，然后采用迭代重定位技术，通过将对象从一个簇移到另一个簇优化划分。

将文档集合 $D=\{d_1,d_2,\cdots,d_n\}$ 水平地分割为若干类，具体过程为：

（1）确定生成的类的数目 k；

（2）按照某种原则生成 k 个聚类中心作为聚类的种子 $S=\{s_1,s_2,\cdots,s_k\}$；

（3）对 D 中的每个文档 d_i，依次计算它与各个种子 s_i 的相似度 $\mathrm{sim}(d_i,s_i)$；

（4）选取具有最大的相似度的种子 $\arg\max \mathrm{sim}(d_i,s_j)$，将 d_i 归入以 s_i 为聚类中心的类 c_i，从而得到 D 的一个聚类 $C=\{c_1,c_2,\cdots,c_k\}$；

(5) 重复步骤(2)～(4),直到获得较为稳定的聚类结果。

该方法运行速度快,但是必须事先确定 k 值,且种子选取的好坏对聚类有较大的影响。常见的平面划分法包括 k-均值和模糊 c-均值。

目前,学术界对于 Web 聚类并没有统一的标准的评价方法,已有的评价方法多借鉴信息检索或文本分类等的评价方法。

9.7 Web 分类

Web 分类是指按照预先定义的分类体系,将 Web 文档集合的每个文档归入某一类别。这样,用户不但能够方便浏览文档,而且可以通过限制搜索范围使文档的查找更为容易。目前,Yahoo 仍然是通过人工对 Web 文档进行分类,这大大限制了其索引页面的数目和覆盖范围。可以说,研究 Web 分类有着广泛的商业前景和应用价值。

文本特征指的是关于文本的元数据,分为描述性特征(例如文本的名称、日期、大小和类型等)以及语义性特征(例如文档的作者、机构、标题、内容等)。对于内容这个难以表示的特征,首先要找到一种能够被计算机所处理的表示方法。

向量空间模型是近年来应用较多的并且效果较好的方法之一。在该模型中,文档被看作是由一组正交词条向量组成的向量空间,每个文档表示为其中的一个规范化特征向量,用文档中出现的所有单词表示其内容特征。预处理的过程首先排除出现频率高但是含义虚泛的词语,例如英文中的 a、the、each、for,汉语中的"地、得、的、这、虽然"等;然后排除那些在文档集合中出现频率很低的单字;在英文中还可以去除前缀,找到词根,如 walker、walking 和 walked 都可以是同一个词 walk。

Web 分类是有指导的机器学习,即利用预定义的分类类别和训练样本集指导学习,预测待分类样本的类别。

从数学角度而言,Web 分类定义为设文档集 $D=\{d_1,d_2,\cdots,d_j\}$,类集 $C=\{c_1,c_2,\cdots,c_j\}$,确定任意一个元组 $<d_j,c_i>$ 映射到集合 $\{T,F\}$ 上的值,故 Web 分类器本质上是一个函数 $D\times C\to\{T,F\}$。

广义而言,分类是数据挖掘的一种方法,但与传统的数据挖掘不同的是,Web 分类的对象是半结构化或非结构化的。目前 Web 分类最普遍的实现方法是将 Web 文档结构化后,再运用传统的分类算法。

Web 分类过程如图 9.14 所示,首先对 Web 文档进行预处理,将文档用模型表示,并进行特征提取;然后构造并训练分类器;最后利用分类器对新的 Web 文档进行分类。

Web 分类基本上都需要经过训练和测试两个阶段,按照 Web 分类的定义,训练阶段就是寻找映射函数的过程,即形成分类器;测试阶段是利用训练生成的模式,自动完成分类。

在 Web 文档建模的基础上,根据文档表示模型的区别,可以将 Web 分类算法分成基于特征独立性算法和基于特征依赖性算法。基于特征独立性算法忽略了文档内词汇或短语之间的语义关系,文档被表现为分量间关系独立的向量,主要利用数学统计方法将 Web 分类问题转换为数学分析,包括相似度模型、概率模型、线性模型、非线性模型和组合模型等。

在相似度模型中,一种方法是计算文档与代表某一文档类别的中心向量之间的相似度

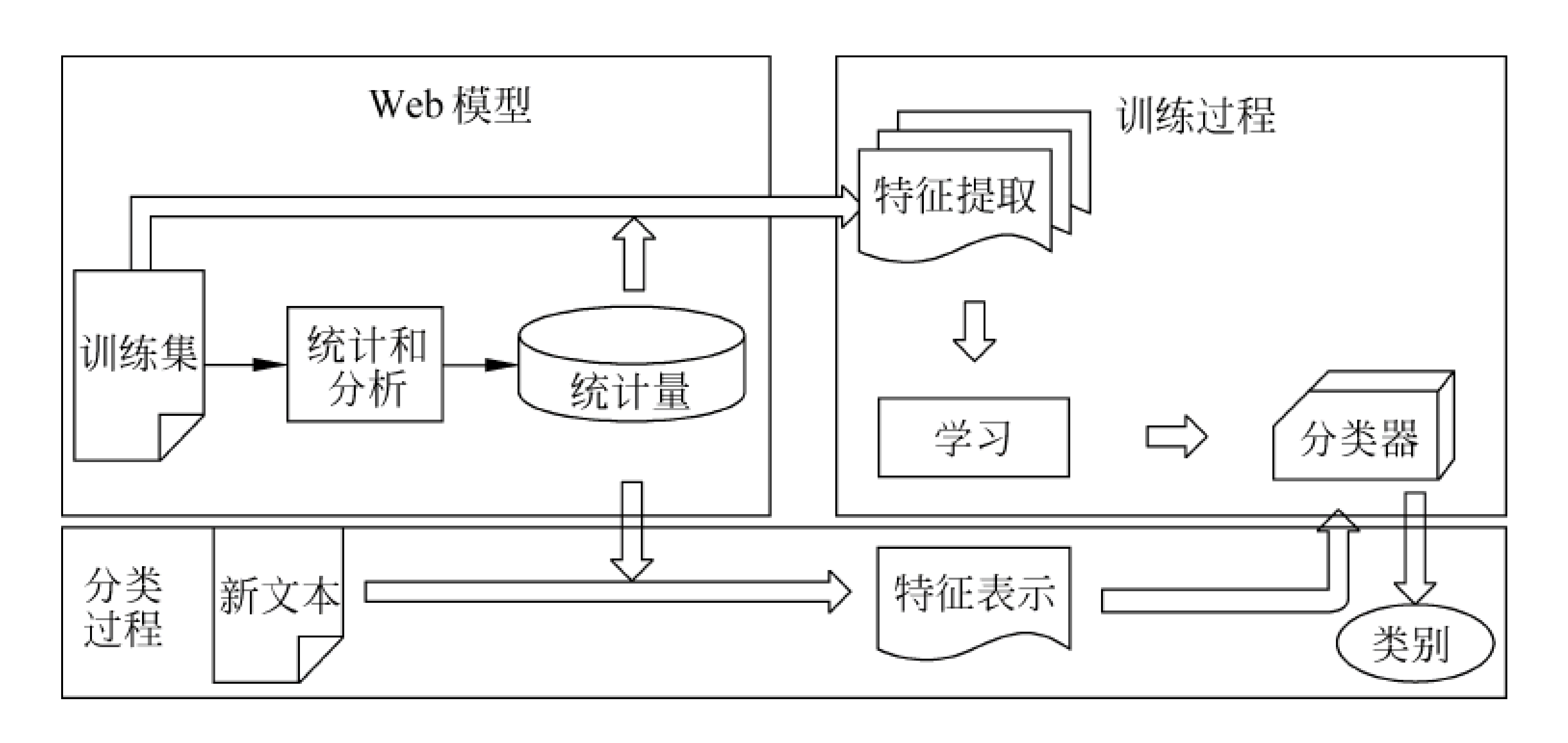


图 9.14　Web 分类过程

$\mathrm{sim}(d_k, c_i)$。其中类别的中心向量是根据测试文本统计计算的估计值。类的中心向量的计算包括算术平均、频率的加权平均和 Rocchio 公式等，相似度的计算方法有多种，常用的包括：

(1) 词条重复率，即只考虑两个特征向量中所包含的词条的重复程度，即：

$$\mathrm{sim}(d_k, c_i) = \frac{n_I(d_k, c_i)}{n_Y(d_k, c_i)} \tag{9.21}$$

其中，$n_I(d_k, c_i)$是 $V(d_k)$和 $V(c_i)$具有的相同词条数目，$n_Y(d_k, c_i)$是 $V(d_k)$和 $V(c_i)$具有的所有词条数目。

(2) 夹角余弦，即考虑两个特征向量之间夹角的余弦，即：

$$\mathrm{sim}(d_k, c_i) = \frac{V(d_k) \cdot V(c_i)}{\beta V(d_k)\beta \times \beta V(C_i)\beta} \tag{9.22}$$

其中，$V(d_k) \cdot V(c_i)$为标准向量点积。

(3) 欧几里得距离，即两个向量的空间距离，即：

$$\sqrt{\sum_{j=1}^{n} (d_{kj} - c_{ij})^2} \tag{9.23}$$

另一种方法不需要建立描述文档类别的中心向量，而是依赖于测试文档与训练文档之间的相似度。典型算法是 k 近邻算法，算法的主要思路是计算训练集与测试文档距离最近的 k 个文档(即 k 个近邻)，根据这 k 个文档的类别判定待分类文档的类别。

9.7.1　朴素贝叶斯

朴素贝叶斯(Naive Bayes，NB)是概率模型的典型算法，其主要思想是基于贝叶斯假设，即文档中的词汇在确定文档类别的作用上相互独立。它首先计算特征词属于每个类别的先验概率，根据特征词的先验概率计算该文档属于每个类别的后验概率，最后取后验概率最大的类别作为分类结果。很多学者对朴素贝叶斯算法进行了改进，如增强型朴素贝叶斯算法、与潜在语义索引结合的贝叶斯方法以及贝叶斯层次分类等。

若文档向量的分量为相应的词在文档中出现的频度(即 TF 表示法)，则采用该方法表示的文档属于 c 类文档的概率为：

$$P(c/\text{Doc}) = \frac{P(c)\prod F_j \in VP(F_j/c)\text{TF}(F_j,\text{Doc})}{\sum_i P(c_i)\prod F_j \in VP(F_l/c_i)\text{TF}(F_i,\text{Doc})} \tag{9.24}$$

$$P(F_j/c) = \frac{1+\text{TF}(F_j,c)}{|V|+\sum_i \text{TF}(F_i,c)} \tag{9.25}$$

其中,$P(c)$为一个文档属于类的概率;$P(F_j/c)$是对 c 类文档中特征 F_j 出现的条件概率的拉普拉斯概率估计;$\text{TF}(F_j,c)$是 c 类文档中特征 F_j 出现的频度;$|V|$为单字辞典集的大小,等于文档表示中所包含的不同特征的总数目;$\text{TF}(F_j,\text{Doc})$是在文档中特征 F_j 出现的频度。

虽然设定条件独立性假设对词汇在文档中出现不是很适合,但 NB 仍是一种有效的方法。

朴素贝叶斯算法的步骤如下:

```
Learn_Native_Bayes(Docs,V)
(1) 收集 Docs 中所有词汇
(2) vocabulary←Docs 中文本文档出现的所有词汇集合
(3) 计算概率 P(Vj)和 P(wk|Vj)
    (3.1) 对 V 分类集中的每一个目标值 Vj,有 docs←Docs 中类标签的 Vj的文档子集;
              P(Vj)←|docs|/|Docs|      n←在 docs 中不同词汇的总数
    (3.2) 对 vocabulary 中每个词汇 wk,有 nk←词汇 wk 在 docsj 中出现的次数 P(wk|Vj)←
             nk + 1
        ──────────────
        n + |vocabulary|
```

贝叶斯分类器的特点如下:

(1) 贝叶斯分类并不把一个对象绝对地指派给某一类,而是通过计算得出属于某一类的概率,具有最大概率的类便是该对象所属的类。

(2) 一般情况下在贝叶斯分类中所有的属性都潜在地起作用,即并不是一个或几个属性决定分类,而是所有的属性都参与分类。

(3) 贝叶斯分类对象的属性可以是离散的、连续的,也可以是混合的。

贝叶斯定理给出了最小化误差的最优化方法,可用于分类和预测。理论上很完美,但在实际中,并不能直接利用,因为需要知道数据的确切分布概率,而实际上并不能确切地给出。因此在很多 Web 分类算法中都会做出某种假设以逼近贝叶斯定理的条件独立性假设。

9.7.2 支持向量机

支持向量机(Support Vector Machines,SVM)是对结构风险最小化原则的近似,该算法的主要思想是在给定的训练集上,作一个超平面的线性划分,将分类问题转化为一个寻找空间最优平面的问题,再次转化成一个二次规划问题。理由是如果所有的向量都能够被某个超平面正确划分,并且各类向量与超平面的最小距离最大化(即边缘最大化),则该平面为最优超平面,距离平面最近的异类向量为支持向量,一组支持向量可以唯一确定一个超平面。

9.7.3 评价

通常地，评价 Web 分类准确率有保留和交叉纠错两种方法，它们都假定待预测样本和训练集样本具有同样的分布。

(1) 保留(Holdout)：数据集的一部分(通常是 2/3)作为训练集，剩余部分用作测试集。利用训练集构造分类器，然后使用这个分类器对测试集进行分类，即评估分类器的准确率。

虽然这种方法速度快，但由于仅使用 2/3 的数据构造分类器，并没有充分利用所有的数据样本进行学习训练。如果使用所有的数据，那么可能构造出更精确的分类器。

(2) 交叉纠错(Cross Validation)：数据集被划分为 k 个没有交叉的子集，所有子集的大小大致相同。分类器训练和测试 k 次；每次分类器使用一个子集的剩余数据作为训练集，然后在该子集上进行测试。最终取所有准确率的平均值作为评估结果。

交叉纠错方法可以重复执行多次，对于一个 t 次 k 分的交叉纠错法，$k\times t$ 个分类器被构造和评估，这意味着交叉纠错法的时间是分类器构造时间的 $k\times t$ 倍。增加重复的次数意味着运行时间的延长和准确率的改善。对 k 值进行调整，减少到 3～5，这样可以缩短运行时间。然而，减小训练集有可能使评估产生较大的偏差。

通常，保留评估法适用于最初的探索性试验，或者数据量多于 5000 的数据集；交叉纠错法用于建立最终的分类器，或者很小的数据集。

Web 分类的评价指标主要包括：

(1) 准确率——模型正确预测未知数据的能力。

(2) 速度——构建和使用模型花费的时间。

(3) 健壮性——有噪声或缺失数据时模型正确分类或预测的能力。

(4) 伸缩性——对于大数据量，有效地构造模型的能力。

(5) 可解释性——学习模型提供的理解和观察的层次。

Web 分类常用的评估指标有分类正确率、查准率(Precision Rate)、查全率(Recall Rate)、F 测度、宏平均和微平均等。其中，分类正确率 A 定义为所有正确分类的样本数与整个测试集样本数的比。查准率和查全率又称为精确率和召回率，是信息检索的评估指标，同样也适用于 Web 分类。查准率是指分类器判定的属于类别 C_i 的所有文档中，确实属于类别 C_i 的文档所占的比例，即：

$$\mathrm{Pr} = \text{实际正确分类的文档数} / \text{分类器分为类 } C_i \text{ 的文档总数}$$

查全率是指原本属于类别 C_i 的所有文档中，分类器正确判定的文档所占的比例，即：

$$\mathrm{Re}=\text{实际正确分类的文档数}/\text{原本属于类 } C_i \text{ 的文档总数}$$

微平均计算所有类别中正确分类和错误分类的样本总数，再求查准率和查全率；宏平均首先计算各个类别的查准率和查全率，然后取算术平均。目前，关于哪种评价方法好还没有定论。

F 测度是由 C. J. Van Rijsbergen 提出，其定义为：

$$F = \frac{2}{\frac{1}{r}+\frac{1}{p}} = \frac{2rp}{r+p} \tag{9.26}$$

其中,r 是查全率,p 是查准率,F 通过赋予查准率和查全率相同的权重平衡对两者的评价。

实际中,影响 Web 分类器准确率的主要因素包括:

(1)训练集的规模。因为利用训练集对分类器进行学习和训练,所以训练集规模越大,分类器性能越可靠。然而,训练集规模越大,构造分类器所需时间就越长。分类准确率随训练集规模的增大而提高。

(2)属性的数量。更多的属性对于分类器而言意味着计算更多的组合,计算时间更长。有时随机关系会将分类器引入歧途,导致可能构造的分类器准确率不高。因此,如果可通过常识分析和判断某个属性与分类无关,则删除。

(3)属性中的信息。有时分类器无法从属性中获取足够的特征进行分类和预测,如试图根据某人眼睛的颜色预测其收入,可考虑引入其他属性,如职业、每周工作小时数和年龄等,以提高准确率。

(4)待测样本的分布。如果待测样本不同于训练集的样本分布,则准确率可能很低。例如利用家用型轿车的训练集构造的分类器,试图用它对运动型轿车进行分类可能没有什么意义,因为样本特征分布可能差别很大。

第 10 章　数据挖掘实例

本章主要介绍如何运用数据挖掘的各种算法和工具，解决实际问题，并且对客户细分、重入网识别、虚开欺诈识别和 WAP 日志挖掘等应用实例进行详细介绍。

10.1　客户细分

在过去国内电信市场竞争相对平缓的环境下，传统的大众化营销模式是成功的，为企业带来了数量庞大的客户群和巨大的经济和社会效益，但随着市场竞争环境的不断演变，竞争越来越白热化，电信产品越来越丰富，价格战不断升级，尤其是面向中高端客户采用传统的大众化营销模式已经不能达到预期目标，无法满足新的市场需求。在这种形势下，需要探索一种有别于传统大众化的更有效的新型营销模式，有针对性地吸引长期、稳定且优质的客户群，以达到更高的产品投资回报率，提高市场占有率，为电信企业创造更高的价值和利润，因此客户细分应运而生。客户细分是在充分了解客户的基础上，通过客户使用行为特征、消费行为特征和自然属性等，区别不同的客户群，以实现针对性营销。

10.1.1　定义

1. 客户生命周期

客户生命周期(又称客户关系生命周期)是指客户关系随时间变化的发展轨迹，即从一个客户开始对企业进行了解或企业欲发展某一客户开始，直到客户与企业的业务关系完全终止且与之相关的事宜完全处理完毕的时间间隔。它直观地揭示了客户关系发展从一种状态向另一种状态迁移的特征。客户生命周期是企业产品生命周期的演变。对企业而言，客户生命周期要比企业某个产品的生命周期更为重要。

根据电信行业客户关系的特点，其客户生命周期可分为五个阶段，分别是识别期，即客户关系的建立阶段；成长期，即客户关系的加强阶段；稳定期，即客户关系的维持稳定阶段；预警期，即客户关系的挽留阶段；离网期，即客户关系的破裂或恢复阶段，如图 10.1 所示。

对于客户生命周期的不同阶段，客户价值与企业的投入都大不相同，企业采取的策略也有所不同。识别期企业需要主动地发现可能建立客户关系的潜在客户，从广大消费群体中挖掘目标客户；成长期企业需要甄别客户类型，挖掘有价值客户，采取客户关系提升策略；稳定期企业需要维系客户关系；预警期企业需要发现衰退迹象，判断客户关系是否值得保持，采取挽留或终止策略；离网期企业需要采取客户关系恢复策略。在明确客户生命周期的基础上，电信运营商需要对客户进行全生命周期管理。客户全生命周期管理是指在假设企业具备生产有市场潜力的产品和服务能力的情况下，如何从广大消费群体中发现目标客户，以及围绕着目标客户关系的建立、发展、成熟和衰退这一生命周期，根据客户关系所处的不同阶段，采用相应的组合策略，对目标客户资源进行动态管理，以期实现企业和客户长期

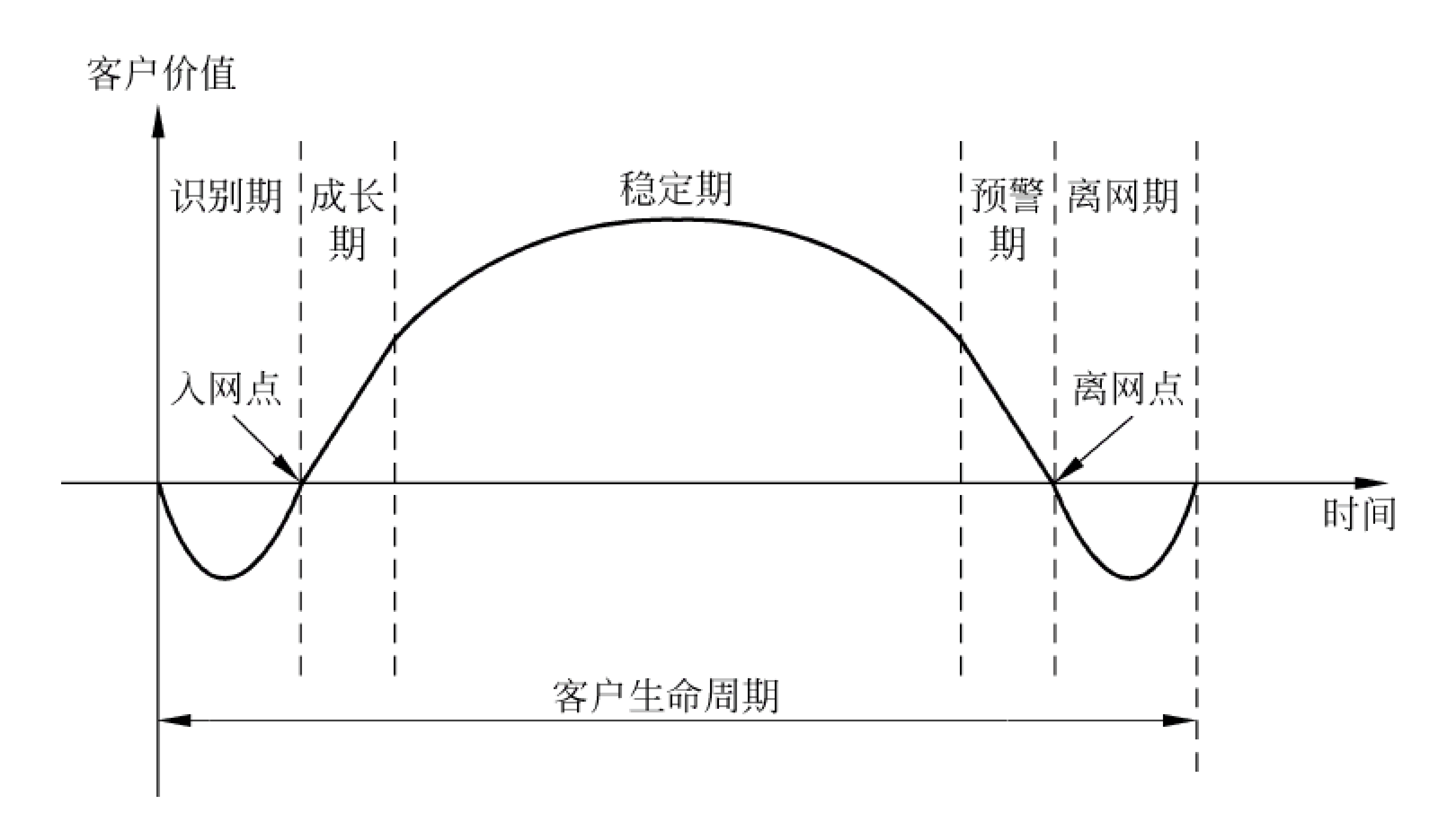


图 10.1　电信客户生命周期

的价值互动,最大化长期互动关系的效用,达到客户与企业的双赢。客户全生命周期管理开始于企业潜在客户的识别,终止于企业与客户关系的破裂。

客户生命周期可以根据客户使用行为、消费行为和客户价值等几大类指标进行划分。

通过对客户特征全面而深入的分析,细分潜在的客户群,进一步识别客户需求,采用特定的营销手段吸引客户注意,使客户知晓企业及企业提供的产品或服务。在持续认知的基础上,客户开始考虑是否使用该企业的产品或服务以满足自身需求。通过对产品的综合评价,客户决定是购买该企业的产品或服务,还是购买竞争对手的产品或服务。一旦客户决定购买该企业的产品或服务,实现了第一次购买,潜在客户就成为实际客户。如果运营商能实现有效的维系,鼓励存量客户购买数量更多、价值更高的服务,客户不断地选择购买该企业的产品和服务,则客户关系得以长期延续。一旦发现客户对该企业的感知度或价值水平下跌而可能流失,则实施有效的挽留,以降低因有价值的客户不再光顾而产生的流失;同时可以终止没有盈利能力的、停止发展的或者不令人满意的客户关系,取而代之的是能够更好地与企业的利润、成长和定位相匹配的客户。

针对电信客户生命周期各阶段的不同特点,运营商可以采取不同的策略预防和控制客户流失,如表 10.1 所示。

表 10.1　电信客户生命周期不同阶段的客户维系挽留策略

客户生命周期阶段	成　长　期	稳　定　期	预　警　期
策略	提升客户价值	提高客户满意度	流失预警
	长远规划营销方案	设置转网壁垒	欠费管理
	市场区隔	老客户回报	带号转资费

2. 客户价值

客户价值是近年来营销领域研究的热点和难点之一。营销科学研究所(Marketing Science Institute)已经连续几年将客户价值列为优先研究领域。对于客户价值的研究可以从三方面展开,即客户为价值感受主体、企业为价值感受主体以及企业和客户互为价值感受主体和感受客体。

从企业的角度研究客户价值,主要包括两个方面,即客户价值和客户终生价值

(Customer Lifetime Value,CLV)。客户价值是指客户当前所产生的净利润。客户终生价值的定义有多种,其中1985年Barbara Jackson将客户终生价值定义为客户当前以及将来所产生的货币利益的净现值;1994年Jackson将客户终身价值定义为企业预计客户在长期的购买行为中,会对该企业带来未来利润的总现值;Bitran和Mondschein认为客户终生价值是客户在整个生命周期内所产生的净利润的折现值。综上所述,客户终生价值可定义为客户在整个生命周期内各个交易期的利润净现值之和。客户终生价值是企业利润的重要来源,客户终生价值越大,对于企业长远发展越有利。

由客户价值和客户终生价值的定义可知,前者关注客户在某一时间点上的价值表现;后者关注客户在整个客户生命周期的最终价值贡献。随着对客户终生价值研究的深入,很自然地将客户终生价值和客户生命周期两者紧密联系起来。客户生命周期中稳定期越长,客户价值折现年限越长,客户价值越高。

客户价值计算

客户价值的含义可以从用客户和企业两个方面分析。从客户的角度而言,客户价值是指客户感知价值,可利用1995年纽曼提出的客户价值特性/成本模型加以说明,如图10.2所示。

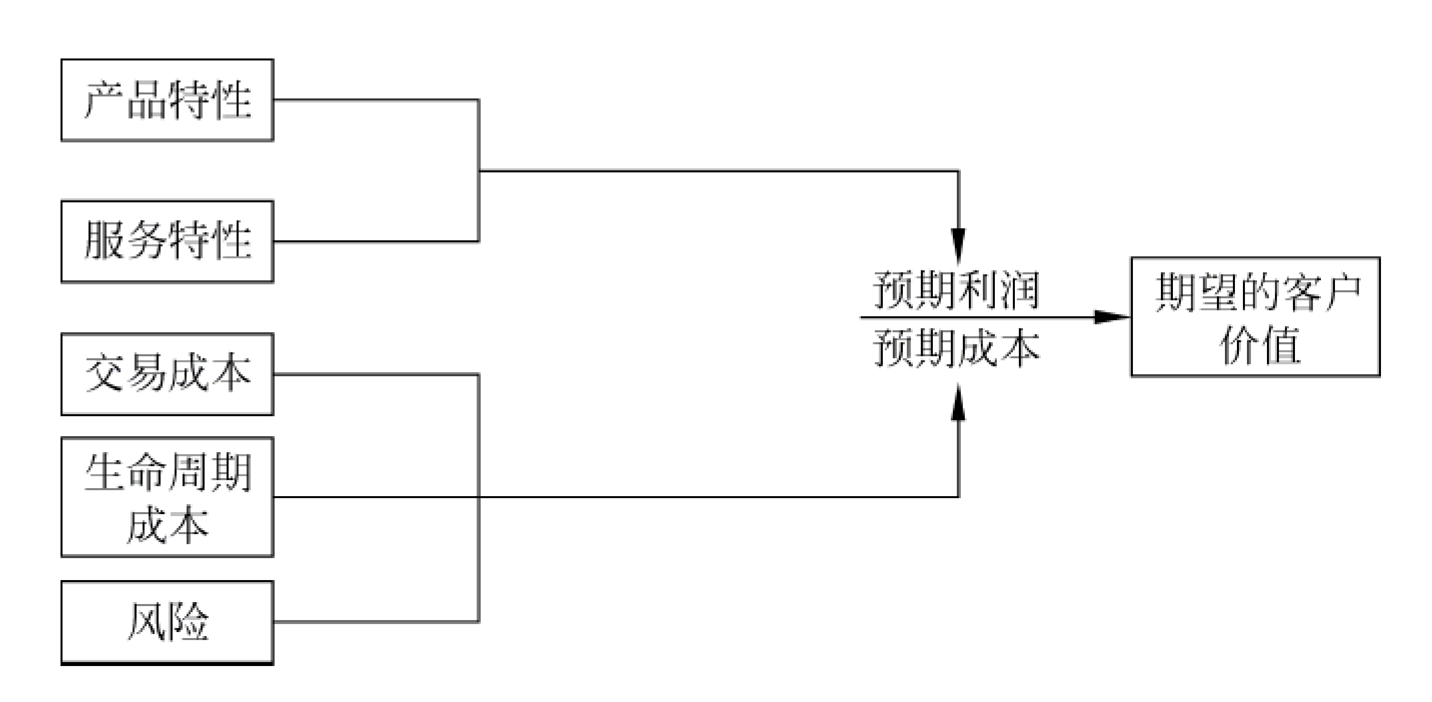


图10.2　客户价值特性/成本模型

从企业的角度而言,客户价值是指客户终生价值,通常由历史价值、当前价值和潜在价值三部分构成。历史价值是指到目前为止已经实现的客户价值;当前价值是指如果客户当前行为模式不发生改变,在将来会给企业带来的客户价值;潜在价值是指如果企业通过有效地交叉销售、调动客户购买积极性或客户向别人推荐产品和服务等,从而可能增加的客户价值。

企业在评价客户是否具有价值时,不仅要参考客户历史价值的表现,更重要的是预测其当前价值和潜在价值的表现。客户历史价值是指已经带来的利润,是企业感知客户价值的一个重要方面;客户当前价值决定了企业当前的盈利水平;客户潜在价值关系到企业的长远利润,是直接影响企业是否继续投资于该客户的一个重要因素。

根据客户终生价值的定义,1995年Angus Jenkison给出其通用的计算公式,即:

$$V = \sum_{n=0}^{T} (R_n - C_n) \times (1 + i)^{-n} \tag{10.1}$$

其中,T为客户的生命周期,R_n为客户在第n年给企业带来的收益(包括货币因素和非货币因素),C_n为企业为其投入的成本,i为银行贴现率。对于客户终生价值的计算,企业可

变的投入成本是影响客户终生价值的重要变量,而客户生命周期时间是一个关键参数。

对于历史价值、当前价值和潜在价值都可以通过直接计算和指标评价两种方法得到。直接计算是指通过严格的数学计算得到具体数值;指标评价是指通过相关指标的评价间接获得评价值。

参照客户终生价值并结合电信客户自身特点给出了一种客户历史价值、当前价值、潜在价值和总体价值的计算方法,分别如式(10.2)～式(10.5)所示。

客户历史价值为

$$V_h = R(t) \times \text{ARPU} - C_i - C_p \tag{10.2}$$

其中,$R(t)$是客户在网时长,C_i 是均摊成本,C_p 是个人维系成本。

客户当前价值

$$V_c = F3 \times (F3/\text{ARPU}) \times K \times B \tag{10.3}$$

其中,$F3$ 是最近三个月应缴费用总额;K 为价值成色,是本地通话与长途和漫游话费的比值,反映话务结构和结算成本;B 为价值爆发力,是最近五个月最高 ARPU 值与 ARPU 均值的比值。

客户潜在价值

$$V_p = \sum \text{ARPU} \times (1 - L(t)) + M \tag{10.4}$$

其中,$L(t)$是客户在未来第 t 个月的流失概率,M 是客户的协议剩余金额。

客户总体价值

$$\begin{aligned} V = & 0.25 \times (V_h - V_{\text{havg}})/\text{SD}(V_h) + 0.5 \times (V_c - V_{\text{cavg}})/SD(V_c) \\ & + 0.25 \times (V_p - V_{\text{pavg}})/\text{SD}(V_p) \end{aligned} \tag{10.5}$$

客户总体价值是客户历史、当前和潜在价值标准化后的加权平均,其中 V_{havg}、V_{cavg} 和 V_{pavg}分别是客户历史价值、当前价值和潜在价值的均值;$\text{SD}(V_h)$、$\text{SD}(V_c)$和 $\text{SD}(V_p)$分别是客户历史价值、当前价值和潜在价值的标准差。

10.1.2 数据准备

为了维系挽留即将到期的 CDMA 合约用户,针对不同客户群的特征实施客户细分,并结合电信公司市场部门的营销策略提供个性化服务,减少客户流失。本例选取某市 2007 年 6～8 月即将到期的 CDMA 合约用户资料,以及最近连续六个月的通话详单和出账费用等相关数据作为分析对象,如表 10.2 所示。

表 10.2 某市 2007 年 6～8 月即将到期的 CDMA 合约用户数量

合约到期的期限(年/月)	用户数	
	在网	正常出账
2007/06	745	238
2007/07	1545	1078
2007/08	6135	3934
合计	8425	5250

由表 10.2 可知，2007 年 6～8 月即将到期的 CDMA 合约用户数共计 8425，正常出账的用户数 5250。对于这部分用户还需要进行数据清洗，以剔除下列无效用户：

(1) 去除入网渠道为“员工”的记录，因为内部员工并非研究对象，属于噪声数据。

(2) 去除孤立点。在进行客户价值聚类时，发现“当月平均总消费”为 1073 元、1060 元和 956 元的三个用户记录使得聚类中心发生严重偏移，去除后聚类效果良好。

(3) 在进行消费行为聚类时，发现“平均套餐优惠费”中存在超大值，去除其中个别的离群值。

(4) 检测某项消费额为负值的用户，无此类记录存在，故应去除。

(5) 在进行消费行为聚类的变量选择时，由于漫游一项只有漫游计费次数，而其他是以通话次数来衡量，随机抽取 20%的数据验证平均长途通话次数与平均长途计费次数大致呈正比关系，因此把漫游计费次数也作为聚类分析的一个变量。

去除上述“噪声”后，针对 2007 年 6～8 月即将到期的有效的 CDMA 合约用户 4824 进行客户细分，并采用 Z Score 方法进行数据标准化。

11.1.3 建模过程

客户细分的建模过程如下：

(1) 利用 K-Means 聚类算法，根据客户价值细分为高端、中高端、中端和低端四类客户群，如表 10.3 所示。

表 10.3 客户价值的聚类结果

类别	客户价值（元）	出账费用明细			用户数	特征描述
高端用户	565.39	主要费用项	金额(元)	占比	108	以漫游和市话为主
		漫游费	267.37	47.3%		
		本地话费	234.89	41.5%		
		长途话费	37.94	6.7%		
		增值费	13.99	2.5%		
		短信费	11.2	2%		
中高端客户	279.96	主要费用项	金额(元)	占比	569	以市话和漫游为主
		本地话费	132.95	47.5%		
		漫游费	101.97	36.4%		
		长途话费	24.43	8.7%		
		增值费	12.64	4.5%		
		短信费	7.97	2.9%		
中端客户	145.69	主要费用项	金额(元)	占比	541	以市话为主和漫游为辅
		本地话费	71.76	49.3%		
		漫游费	47.43	32.6%		
		长途话费	12.47	8.6%		
		增值费	8.75	6%		
		短信费	5.28	3.5%		

续表

类　别	客户价值(元)	出账费用明细			用户数	特 征 描 述
低端客户	29.94	主要费用项	金额(元)	占比	3606	以市话和短信为主
		本地话费	15.69	52.4%		
		短信费	4.53	15.1%		
		增值费	3.73	12.5%		
		长途话费	3.55	11.9%		
		漫游费	2.44	8.1%		
合计	4824					

由图 10.3 可以看出聚类效果较为显著,绝大多数很集中,不存在极值,只有个别的孤立点,且它们距离聚类中心并不远。

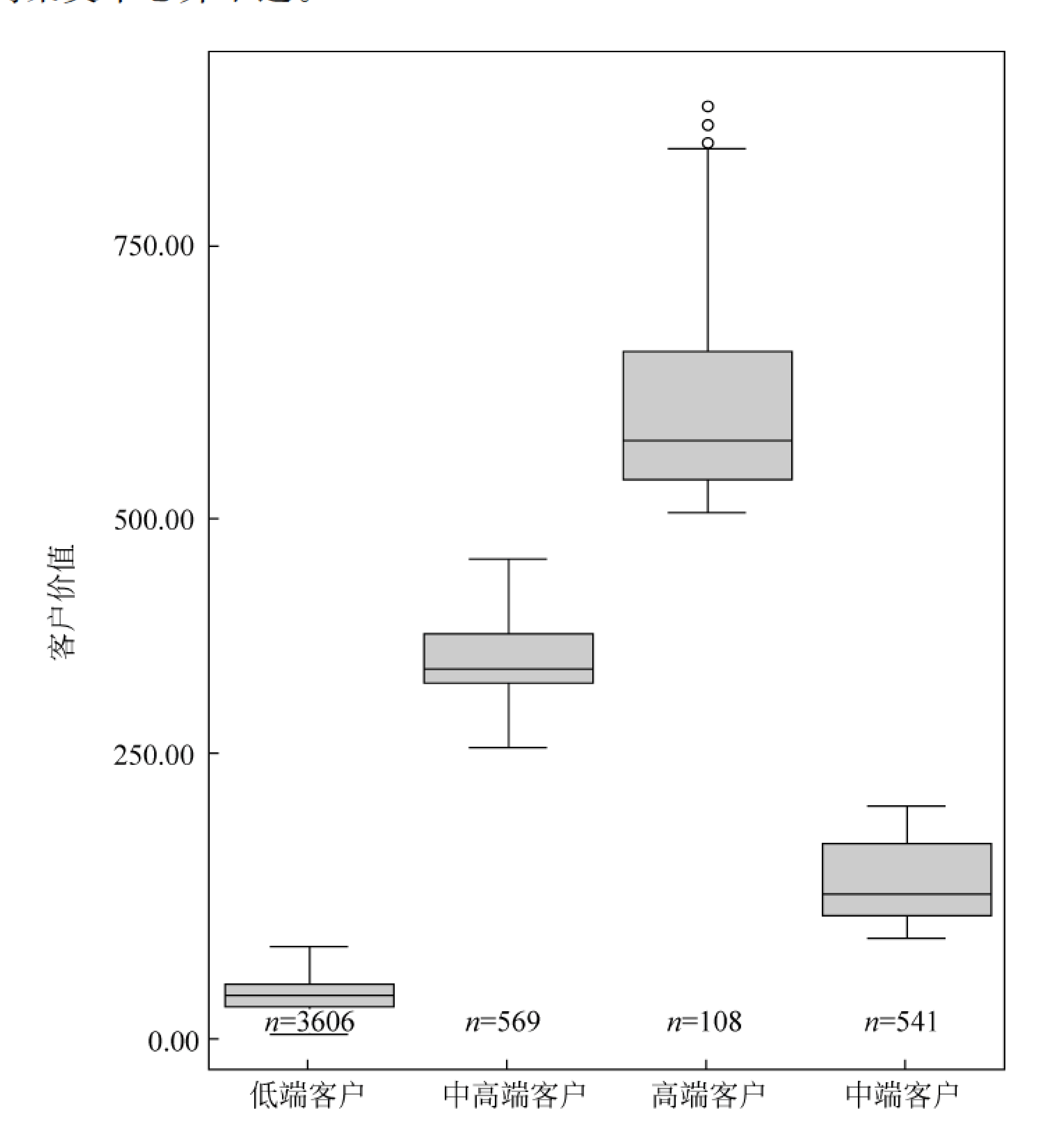


图 10.3　四种客户群的客户价值分布

(2) 利用 K-means 聚类算法,根据客户消费行为,如月均基本通话费、月均长途费、月均漫游费和月均短信费等指标聚类为节约型、时尚型、电话型、长途型、短信型和未知型六类客户群,如表 10.4 和图 10.4 所示。

表 10.4　客户消费行为的聚类结果

序号	类 别 名 称	用户数
1	节约型	3703
2	时尚型(以增值、长途、漫游为主)	20
3	市话型(以市话、长途、漫游为主)	395
4	长途型	189
5	短信型	185
6	未知型(其他费用高)	332
合计	4824	

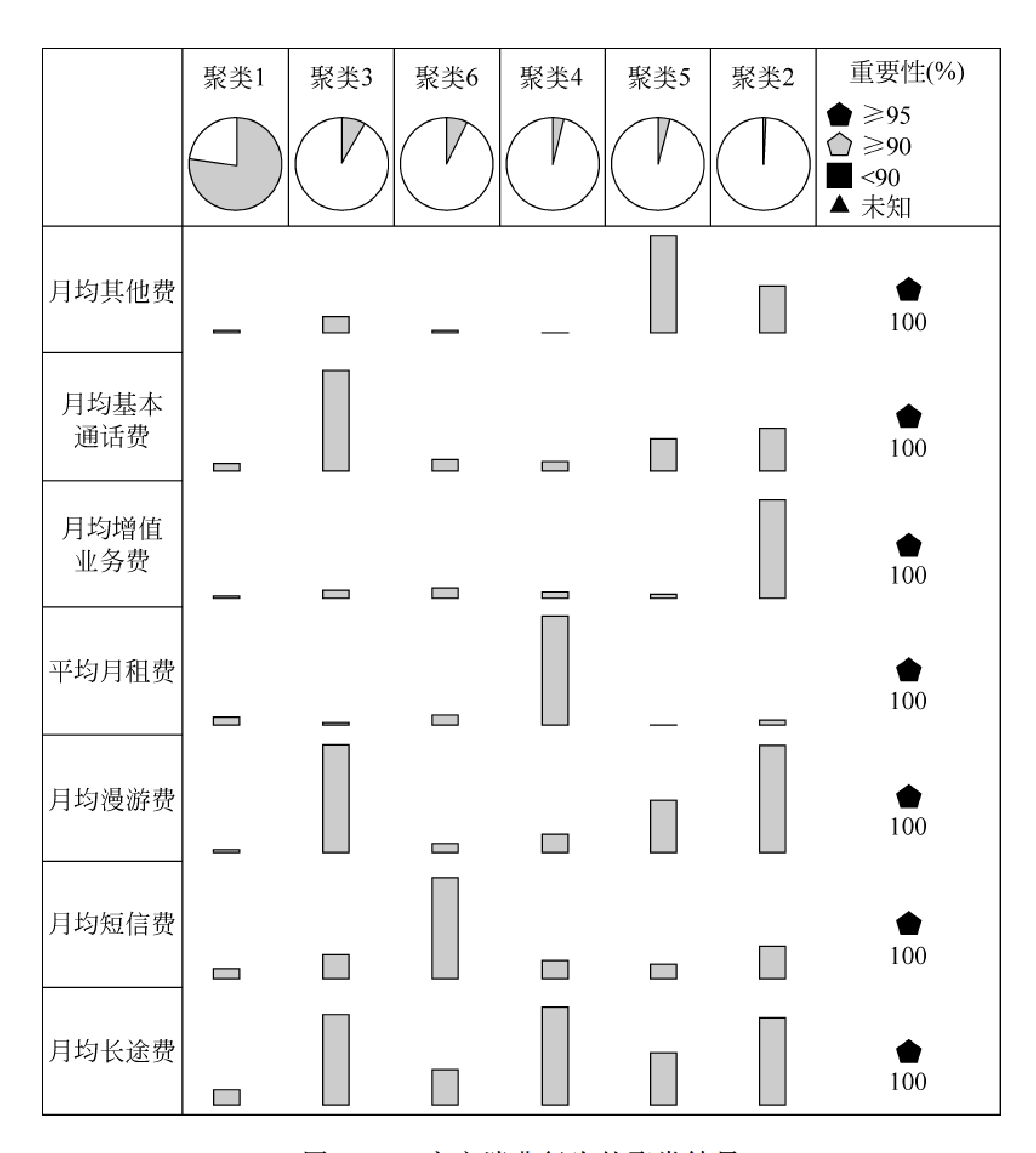


图 10.4　客户消费行为的聚类结果

(3) 利用 K-means 聚类算法,根据客户通话行为特征,如月均本地通话次数、月均主叫通话次数、月均被叫通话次数、月均长途通话次数、月均 IP 通话次数、月均漫游计费次数和月均假期通话次数指标聚类为极少型、假期型、长途漫游型、普通型和 IP 电话型五类客户群,如表 10.5 和图 10.5 所示。

表 10.5 客户通话行为的聚类结果

序号	类别名称	用户数
1	极少型	3561
2	假期型	114
3	长途漫游型	249
4	普通型	806
5	IP 电话型	94
合计	4824	

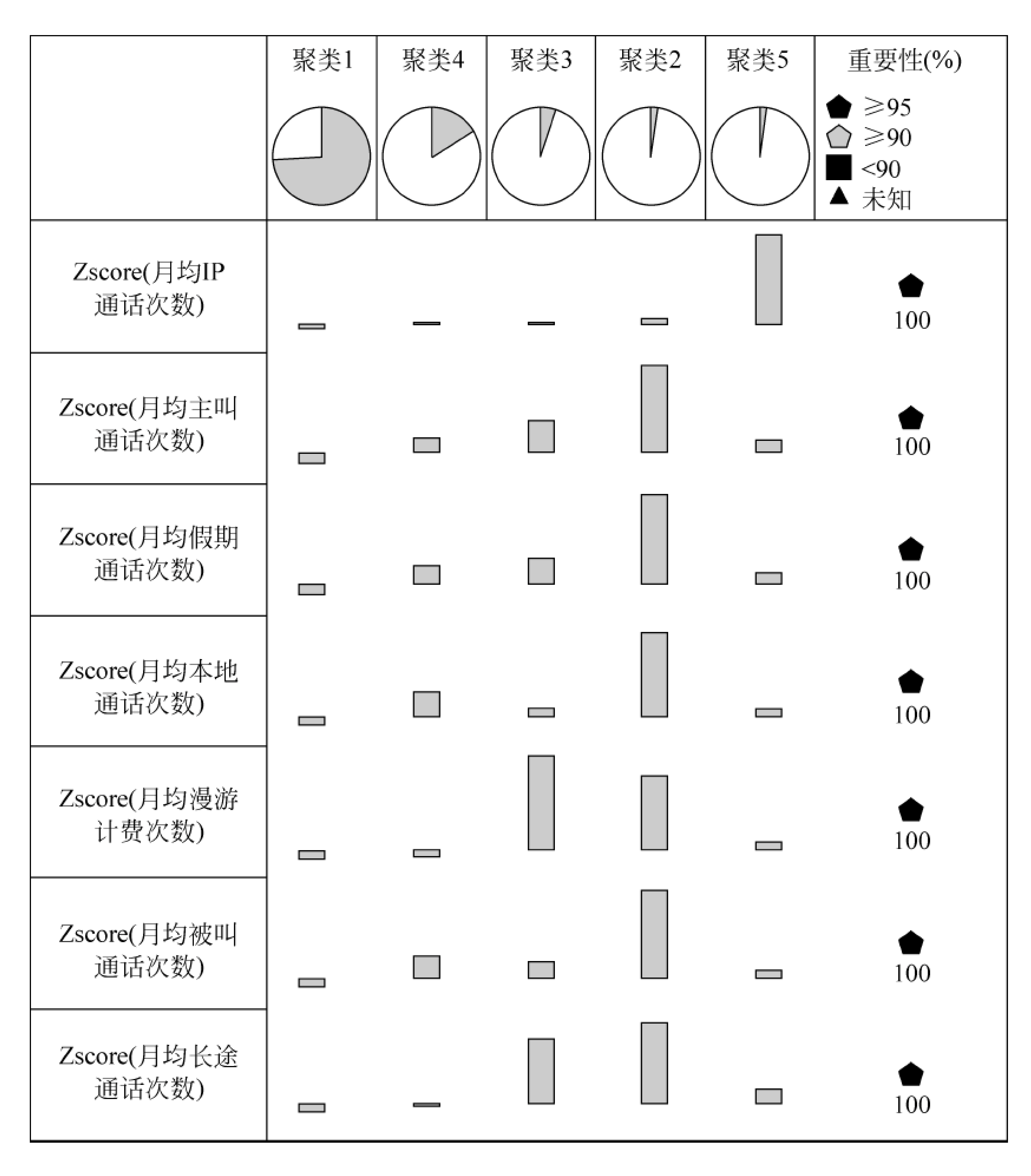


图 10.5 客户通话行为的聚类结果

(4) 汇总上述聚类结果,如表 10.6 所示。电信公司市场营销部门可以根据不同客户群的价值贡献、消费行为特征和通话行为特征推荐适合的优惠套餐,实现个性化服务。

表 10.6 聚类结果的汇总

客户价值	消费行为		通话行为					合计
			极少型	假期型	长途漫游型	普通型	IP 电话型	
低端客户	节约型	用户数	2999	1	0	352	19	3371
		消费行为占比	89.0%	0.0%	0.0%	10.4%	0.6%	
		通话行为占比	93.7%	100.0%	0.0%	91.7%	95.0%	
		总占比	83.2%	0.0%	0.0%	9.8%	0.5%	93.5%
	短信型	用户数	202	0	0	32	1	235
		消费行为占比	86.0%	0.0%	0.0%	13.6%	0.4%	
		通话行为占比	6.3%	0.0%	0.0%	8.3%	5.0%	
		总占比	5.6%	0.0%	0.0%	0.9%	0.0%	6.5%
	合计	用户数	3201	1	0	384	20	3606
		消费行为占比	88.8%	0.0%	0.0%	10.6%	0.6%	100.0%
		通话行为占比	100.0%	100.0%	0.0%	100.0%	100.0%	100%
		总占比	88.8%	0.0%	0.0%	10.6%	0.6%	100.0%
中高端客户	节约型	用户数	2	1	6	3	3	15
		消费行为占比	13.3%	6.7%	40.0%	20.0%	20.0%	
		通话行为占比	1.6%	2.1%	3.5%	1.6%	7.9%	
		总占比	0.4%	0.2%	1.1%	0.5%	0.5%	2.7%
	时尚型（增值漫游长途）	用户数	4	0	5	2	0	11
		消费行为占比	36.4%	0.0%	45.4%	18.2%	0.0%	
		通话行为占比	3.3%	0.0%	2.9%	1.0%	0.0%	
		总占比	0.7%	0.0%	0.9%	0.4%	0.0%	2.0%
	市话型（市话漫游长途）	用户数	9	21	126	79	21	256
		消费行为占比	3.5%	8.2%	49.2%	30.9%	8.2%	
		通话行为占比	7.4%	43.7%	74.1%	41.4%	55.2%	
		总占比	1.6%	3.7%	22.1%	13.9%	3.7%	45.0%
	长途型	用户数	24	25	1	72	4	126
		消费行为占比	19.1%	19.8%	0.8%	57.1%	3.2%	
		通话行为占比	19.7%	52.1%	0.6%	37.7%	10.5%	
		总占比	4.2%	4.4%	0.2%	12.7%	0.7%	22.2%
	未知型（其他费用高）	用户数	81	0	27	32	5	145
		消费行为占比	55.9%	0.0%	18.6%	22.1%	3.4%	
		通话行为占比	66.4%	0.0%	15.9%	16.7%	13.2%	
		总占比	14.2%	0.0%	4.7%	5.6%	0.9%	25.4%
	短信型	用户数	2	1	5	3	5	16
		消费行为占比	12.5%	6.2%	31.3%	18.7%	31.3%	
		通话行为占比	1.6%	2.1%	3.0%	1.6%	13.2%	
		总占比	0.4%	0.2%	0.9%	0.5%	0.9%	2.9%
	合计	用户数	122	48	170	191	38	569
		消费行为占比	21.4%	8.4%	29.9%	33.6%	6.7%	100.0%
		通话行为占比	100.0%	100.0%	100.0%	100.0%	100.0%	100%
		总占比	21.4%	8.4%	29.9%	33.6%	6.7%	100.0%

续表

客户价值	消费行为		通话行为					合计
			极少型	假期型	长途漫游型	普通型	IP电话型	
高端客户	时尚型(增值漫游长途)	用户数	1	2	5	0	0	8
		消费行为占比	12.5%	25.0%	62.5%	0.0%	0.0%	
		通话行为占比	33.3%	4.3%	10.6%	0.0%	0.0%	
		总占比	0.9%	1.9%	4.6%	0.0%	0.0%	7.4%
	市话型(市话漫游长途)	用户数	0	37	34	4	1	76
		消费行为占比	0.0%	48.7%	44.7%	5.3%	1.3%	
		通话行为占比	0.0%	78.7%	72.3%	44.4%	50.0%	
		总占比	0.0%	34.3%	31.5%	3.7%	0.9%	70.4%
	长途型	用户数	2	8	3	4	0	17
		消费行为占比	11.8%	47.1%	17.6%	23.5%	0.0%	
		通话行为占比	66.7%	17.0%	6.4%	44.4%	0.0%	
		总占比	1.9%	7.4%	2.8%	3.7%	0.0%	15.8%
	未知型(其他费用高)	用户数	0	0	4	0	1	5
		消费行为占比	0.0%	0.0%	80.0%	0.0%	20.0%	
		通话行为占比	0.0%	0.0%	8.5%	0.0%	50.0%	
		总占比	0.0%	0.0%	3.7%	0.0%	0.9%	4.6%
	短信型	用户数	0	0	1	1	0	2
		消费行为占比	0.0%	0.0%	50.0%	50.0%	0.0%	
		通话行为占比	0.0%	0.0%	2.1%	11.1%	0.0%	
		总占比	0.0%	0.0%	0.9%	0.9%	0.0%	1.8%
	合计	用户数	3	47	47	9	2	108
		消费行为占比	2.8%	43.5%	43.5%	8.3%	1.9%	100.0%
		通话行为占比	100.0%	100.0%	100.0%	100.0%	100.0%	100%
		总占比	2.8%	43.5%	43.5%	8.3%	1.9%	100.0%

10.1.4 结果

由上述客户细分可知，低端客户群的消费类型主要集中在节约型(93.5%)和短信型(6.5%)两种；通话行为类型主要集中在极少型(88.8%)和普通型(10.6%)。且其中绝大部分的客户既是节约型又是极少型(83.2%)，此外还有9.8%的客户非常节约，基本限于本地通话；还有5.6%的客户主要使用短信。

中端客户群的节约型和普通型占比最大，短信一族次之。从消费行为特征上看，节约型占比58.6%，短信型占比15.6%；从通话行为特征上看，极少型和普通型分别占44%和41.4%。

中高端客户群的消费行为和通话行为较为分散，多数以市话、长途和漫游通话为主。

高端客户群的市话、长途和漫游费用占比很高。从消费行为特征上看，市话型占比70.4%，而长途型只占15.7%；从通话行为特征上看，假期型和长途漫游型各占43.5%，因此高端客户群的主要价值来源于漫游和长途业务，且在假期客户价值尤其高。

表 10.7 根据消费行为和通话行为聚类的结果

消费行为		通话行为					合计
		极少型	假期型	长途漫游型	普通型	IP电话型	
节约型	用户数	3140	12	25	488	38	3703
	消费行为占比	84.8%	0.3%	0.7%	13.2%	1.0%	
	通话行为占比	88.2%	8.1%	10.0%	60.6%	39.1%	
	总占比	65.2%	0.2%	0.5%	10.2%	0.7%	76.8%
时尚型（增值漫游长途）	用户数	6	2	10	2	0	20
	消费行为占比	30.0%	10.0%	50.0%	10.0%	0.0%	
	通话行为占比	0.2%	1.8%	4.0%	0.2%	0.0%	
	总占比	0.1%	0.0%	0.2%	0.0%	0.0%	0.4%
市话型（市话漫游长途）	用户数	25	61	172	109	28	395
	消费行为占比	6.3%	15.4%	43.6%	27.6%	7.1%	
	通话行为占比	0.7%	55.0%	69.1%	13.5%	30.4%	
	总占比	0.5%	1.3%	3.6%	2.3%	0.6%	8.2%
长途型	用户数	39	37	4	104	5	189
	消费行为占比	20.6%	19.6%	2.1%	55.1%	2.6%	
	通话行为占比	1.1%	33.3%	1.6%	12.9%	5.4%	
	总占比	0.8%	0.8%	0.1%	2.2%	0.1%	3.9%
未知型（其他费用高）	用户数	111	0	32	34	8	185
	消费行为占比	60.0%	0.0%	17.3%	18.4%	4.3%	
	通话行为占比	3.1%	0.0%	12.9%	4.2%	8.7%	
	总占比	2.3%	0.0%	0.7%	0.7%	0.2%	3.8%
短信型	用户数	240	2	6	69	15	332
	消费行为占比	72.3%	0.6%	1.8%	20.8%	4.5%	
	通话行为占比	6.7%	1.8%	2.4%	8.5%	16.3%	
	总占比	5.0%	0.0%	0.1%	1.4%	0.3%	6.9%
合计	用户数	3561	114	249	806	94	4824
	消费行为占比	73.9%	2.3%	5.2%	16.7%	1.9%	100.0%
	通话行为占比	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
	总占比	73.9%	2.3%	5.2%	16.7%	1.9%	100.0%

由表10.7可知，节约型与极少型客户几乎一一对应，包含了少量的普通型；时尚型客户除了长途和漫游费用较高外，还有很大一部分增值业务的费用，这部分客户数量不多，主要集中在长途漫游型；市话型客户，即市话、长途和漫游费用均很高的客户，集中在长途漫游型，还有一部分普通型客户；长途型客户有一半是普通型，还有少量的极少型和假期型用户；未知型客户其他费用极高，且客户数不在少数，60%的客户是极少型，据推测其他费用可能是使用了数字业务导致语音业务很少的客户反而其他费用高；短信型客户大部分来源于节约型，还有20.8%的普通型，与实际情况相符。

从通话行为上看，极少型对应节约型，假期型对应时尚型和长途型，长途漫游型对应市

话型,普通型对应节约型,IP 市话型对应市话型和节约型。

综上所述,某市 2007 年 6～8 月即将到期 CDMA 合约客户细分结果如表 10.8 所示。

表 10.8 某市 2007 年 6～8 月即将到期 CDMA 合约客户细分结果

序号	类别名称	特征描述	用户数	营销策略
1	节省型	短信量大,电话量少	3450	短信优惠
2	均衡型	被叫明显多于主叫,各业务费用均衡	737	被叫优惠,各种业务绑定优惠
3	长途型	长途费用高,几乎无漫游费用	189	长途优惠
4	市话长途漫游型	市话、长途、漫游费用均高	209	市话、长途、漫游包月优惠及绑定优惠
5	增值长途漫游型	用户群小,增值费用高	20	可不考虑或归入第 4 类
6	假期型	假期时各项费用均高	114	针对假期时段的综合优惠

此外,4824 客户中选择各种"校园套餐"的用户数是 2920,占 60.5%,他们大多数为低端客户,其通话行为特征表现为节省型,月均消费小于 50 元;选择各种"商旅套餐"的客户数是 1090,约占 22.6%,他们大多数为高端和中高端客户,其通话行为特征表现为市话长途漫游型或市话型,月均消费大于 370 元。这两类客户群在消费行为特征和通话行为特征方面差异较明显,占 83%以上,建议进行重点维挽。分析中还可看出漫游业务较多的客户长途业务也多,可针对市话长途漫游型和增值长途漫游型推出组合业务,或针对二者共同推出长途漫游绑定业务。

10.2 重入网识别

10.2.1 定义

重入网是指已经拥有或曾经拥有某一家电信运营商的一张 SIM 卡(又称用户识别卡)的情况下,重新在本地或本省其他地区购买新的同一运营商的 SIM 卡,新卡部分或者全部替代旧卡功能。

由于市场竞争日趋激烈,以及增量市场趋于饱和,重入网现象日趋严重,比例也越来越高。重入网造成的卡号资源浪费、欠费等后果直接导致运营商营销成本的增加和业务收入的下滑。因此,控制重入网比例,有效降低重入网带来的运营成本和风险,目前已经成为运营商面临的难题,重入网识别正是在这样的背景下应运而生,其应用价值在于为代理商佣金政策动态调整和绩效考核提供科学依据,提高维系挽留的精确度和维系成本使用的有效性。

10.2.2 数据准备

数据准备主要是选择合适的数据源,整理分析所需的各种数据,包括通话、消费、欠费和客服等,对原始数据进行检查和预处理,生成新的衍生变量,并整理出适合分析使用的数据

宽表。一方面尽量确保数据的完整性；另一方面剔除冗余，减少噪声。

根据业务需求选取某市2008年9月流失的CDMA客户45 537和2008年9～10月两个月新发展的CDMA用户135 638作为分析对象。从ODS系统和业务支撑系统抽取大量通话详单、客户资料和出账等数据，按照客户ID进行汇总和合并，生成一张数据宽表，其中每行代表一个客户，每列代表一个变量。数据宽表如表10.9所示。

表10.9　数据宽表

类别	列　名　称	列　类　型	列　描　述
客户信息	CUST_ID	CHAR(20)	客户ID
	GENDER	CHAR(1)	性别
	AGE	NUMERIC	年龄
	IS_LOC	CHAR(1)	是否本地居民
	NUM_OF_USERS	NUMERIC	客户对应的用户总数
	DATE	DATE	统计日期
	ACC_DATE	DATE	出账日期
用户信息	USER_ID	CHAR(15)	用户ID
	PIN	CHAR(15)	号码
	VIP_TYPE	CHAR(2)	用户价值
	PAY_MODE	CHAR(2)	用户付费方式
	HANDSET_MODEL	CHAR(20)	用户手机类型
	MANUFACTURER	CHAR(20)	用户手机厂家
	IMEI	CHAR(15)	用户手机IMEI串号
	COLLECT_MODE	CHAR(1)	催缴模式
	SVC_ID	CHAR(5)	业务类型
	INNET_LOC	CHAR(4)	入网地区
	DEPT_NO	CHAR(4)	部门编码
	INNET_CHANNEL	CHAR(1)	入网渠道
	SALE_TYPE	CHAR(2)	销售模式
	CALL_LOC	CHAR(4)	当前活动区域
	CALL_LOC_LST	CHAR(4)	上次活动区域
	INNET_AGE	NUMERIC	入网年龄
	INNET_DATE	DATE	入网日期
	Tenure_IN_M	NUMERIC	在网时长(月)
	ARPU	NUMERIC	ARPU
	MOU	NUMERIC	MOU
	ARPM	NUMERIC	ARPM

续表

类别	列　名　称	列　类　型	列　描　述
资费信息	PLAN_INIT	CHAR(5)	初始套餐
	PLAN_LAST	CHAR(5)	最近套餐
	PLAN_CURR	CHAR(5)	当前套餐
	PLAN_CHG_CNT	NUMERIC	套餐变动总数
	CONS_DUE_DATE	DATE	当前承诺到期日
	IS_CONS_OVER	CHAR(1)	是否已经承诺到期
	IS_CONS_DUE	CHAR(1)	是否当月承诺到期
	CONS_TYPE	CHAR(3)	承诺类型
	CONS_CHG	NUMERIC	是否有承诺话费
	HANDSET_FREE	CHAR(20)	承诺是否赠送手机
	HANDSET_PRICE	NUMERIC	承诺赠送手机价值
	DINNER_SPEC	CHAR(5)	特服套餐
	RENT_PACKAGE	NUMERIC	套餐租费
	IS_OVR_LMT	CHAR(1)	是否超过资费套餐定量
	RAT_OVR_LMT	NUMERIC	超过资费套餐定量部分与定量的比例
	TIME_TO_LST_PLN	NUMERIC	距最近一次套餐变更的时间
	PLAN_PRICE	CHAR(1)	套餐价位
	VOC_PACKAGE	NUMERIC	包月话费
	VAS_PACKAGE	NUMERIC	包月特服
	SMS_PACKAGE	NUMERIC	包月短信
	PLAN_UTIL	NUMERIC	套餐使用度(针对定额或包月套餐)
	AVG_SMS_CHG	NUMERIC	平均每条短信费用
	AVG_PEAK_CALL_CHG	NUMERIC	平均每分钟忙时呼出费用
	AVG_OFPK_CALL_CHG	NUMERIC	平均每分钟闲时呼出费用
	AVG_PEAK_CALL_CHG	NUMERIC	平均每分钟忙时呼入费用
	AVG_OFPK_CALL_CHG	NUMERIC	平均每分钟闲时呼入费用
	IS_COMP_PLAN	CHAR(1)	套餐是否针对竞争对手推出
	IS_LV_PLAN	CHAR(1)	套餐是否针对低端人群
	IS_HV_PLAN	CHAR(1)	套餐是否针对高端人群
	IS_LD_PLAN	CHAR(1)	套餐是否针对长途业务优惠
	IS_RM_PLAN	CHAR(1)	套餐是否针对漫游业务优惠
	IS_IN_PLAN	CHAR(1)	套餐是否针对接听优惠
	IS_SMS_PLAN	CHAR(1)	套餐是否针对短信优惠
	IS_VAS_PLAN	CHAR(1)	套餐是否针对增值服务优惠
	HANDSET_BIND	CHAR(20)	套餐是否绑定话机

续表

类别	列　名　称	列　类　型	列　描　述
账户信息	PAY_TYPE	CHAR(2)	缴费渠道
	STATUS_CODE	CHAR(3)	账户状态
	DUE_CHARGE	NUMERIC	当月应缴金额
	CROSS_CHG	NUMERIC	结算费用总额
	ACT_PAID	NUMERIC	当月实缴金额
	RENT_FEE	NUMERIC	租费
	MOBILE_FEE	NUMERIC	本地通话费
	LONG_FEE	NUMERIC	本地国内长途费
	INTLONG_FEE	NUMERIC	本地国际长途费
	ROAM_FEE	NUMERIC	漫游费
	ROAM_INTLONG_FEE	NUMERIC	漫游国际长途费
	ROAM_LONG_FEE	NUMERIC	漫游国内长途费
	DATA_FEE	NUMERIC	数据业务费
	INCREMENT_FEE	NUMERIC	增值业务费
	SMS_FEE	NUMERIC	点对点短信费
	SPECIAL_FEE	NUMERIC	特服业务费
	MON_FEE	NUMERIC	包月费
	OTHER_FEE	NUMERIC	其他费
	SP_FEE	NUMERIC	SP 业务费
	CONSENT_FEE	NUMERIC	最低承诺应收费
	DELQ_CNT_CURR	NUMERIC	本次欠费持续时间
	DELQ_AMT_CURR	NUMERIC	本次欠费总额
行为信息	AVG_PAY	NUMERIC	月均缴费额
	TOTAL_PAY	NUMERIC	缴费总额
	DELQ_CNT_EVER	NUMERIC	累计欠费次数
	DELQ_HALT_CNT_EVER	NUMERIC	累计欠费停机次数
	PIN	CHAR(15)	号码
	CALL_LENGTH	NUMERIC	通话时长
	CALL_CNT	NUMERIC	通话次数
	CALL_CHG	NUMERIC	通话费用
	FREE_LENGTH	NUMERIC	免费通话时长
	FREE_CNT	NUMERIC	免费通话次数
	IN_CALL_LENGTH	NUMERIC	呼入通话时长
	IN_CALL_CNT	NUMERIC	呼入通话次数
	IN_CALL_CHG	NUMERIC	呼入通话费用
	OUT_CALL_LENGTH	NUMERIC	呼出通话时长
	OUT_CALL_CNT	NUMERIC	呼出通话次数
	OUT_CALL_CHG	NUMERIC	呼出通话费用
	IN_FREE_LENGTH	NUMERIC	呼入免费通话时长
	IN_FREE_CNT	NUMERIC	呼入免费通话次数
	OUT_FREE_LENGTH	NUMERIC	呼出免费通话时长
	OUT_FREE_CNT	NUMERIC	呼出免费通话次数
	IN_PK_CALL_LENGTH	NUMERIC	忙时呼入通话时长
	IN_PK_CALL_CNT	NUMERIC	忙时呼入通话次数

续表

类别	列 名 称	列 类 型	列 描 述
	IN_PK_CALL_CHG	NUMERIC	忙时呼入通话费用
	OUT_PK_CALL_LENGTH	NUMERIC	忙时呼出通话时长
	OUT_PK_CALL_CNT	NUMERIC	忙时呼出通话次数
	OUT_PK_CALL_CHG	NUMERIC	忙时呼出通话费用
	IN_OP_CALL_LENGTH	NUMERIC	闲时呼入通话时长
	IN_OP_CALL_CNT	NUMERIC	闲时呼入通话次数
	IN_OP_CALL_CHG	NUMERIC	闲时呼入通话费用
	OUT_OP_CALL_LENGTH	NUMERIC	闲时呼出通话时长
	OUT_OP_CALL_CNT	NUMERIC	闲时呼出通话次数
	OUT_OP_CALL_CHG	NUMERIC	闲时呼出通话费用
	PK_CALL_LENGTH	NUMERIC	忙时通话时长
	PK_CALL_CNT	NUMERIC	忙时通话次数
	PK_CALL_CHG	NUMERIC	忙时通话费用
	OP_CALL_LENGTH	NUMERIC	闲时通话时长
	OP_CALL_CNT	NUMERIC	闲时通话次数
	OP_CALL_CHG	NUMERIC	闲时通话费用
	PK_FREE_LENGTH	NUMERIC	忙时免费通话时长
	PK_FREE_CNT	NUMERIC	忙时免费通话次数
	OP_FREE_LENGTH	NUMERIC	闲时免费通话时长
	OP_FREE_CNT	NUMERIC	闲时免费通话次数
	WK_CALL_LENGTH	NUMERIC	平时通话时长
行为信息	WK_CALL_CNT	NUMERIC	平时通话次数
	WK_CALL_CHG	NUMERIC	平时通话费用
	HL_CALL_LENGTH	NUMERIC	假日通话时长
	HL_CALL_CNT	NUMERIC	假日通话次数
	HL_CALL_CHG	NUMERIC	假日通话费用
	LOC_CALL_LENGTH	NUMERIC	区内通话时长
	LOC_CALL_CNT	NUMERIC	区内通话次数
	LOC_CALL_CHG	NUMERIC	区内通话费用
	DD_CALL_LENGTH	NUMERIC	国内长途通话时长
	DD_CALL_CNT	NUMERIC	国内长途通话次数
	DD_CALL_CHG	NUMERIC	国内长途通话费用
	IDD_CALL_LENGTH	NUMERIC	国际长途通话时长
	IDD_CALL_CNT	NUMERIC	国际长途通话次数
	IDD_CALL_CHG	NUMERIC	国际长途通话费用
	DD_PK_CALL_LENGTH	NUMERIC	国内长途忙时通话时长
	DD_PK_CALL_CNT	NUMERIC	国内长途忙时通话次数
	DD_OP_CALL_LENGTH	NUMERIC	国内长途闲时通话时长
	DD_OP_CALL_CNT	NUMERIC	国内长途闲时通话次数
	DD_WK_CALL_LENGTH	NUMERIC	国内长途平时通话时长
	DD_WK_CALL_CNT	NUMERIC	国内长途平时通话次数
	DD_HL_CALL_LENGTH	NUMERIC	国内长途假日通话时长
	DD_HL_CALL_CNT	NUMERIC	国内长途假日通话次数
	IDD_PK_CALL_LENGTH	NUMERIC	国际长途忙时通话时长

续表

类别	列　名　称	列　类　型	列　描　述
行为信息	IDD_PK_CALL_CNT	NUMERIC	国际长途忙时通话次数
	IDD_OP_CALL_LENGTH	NUMERIC	国际长途闲时通话时长
	IDD_OP_CALL_CNT	NUMERIC	国际长途闲时通话次数
	IDD_WK_CALL_LENGTH	NUMERIC	国际长途平时通话时长
	IDD_WK_CALL_CNT	NUMERIC	国际长途平时通话次数
	IDD_HL_CALL_LENGTH	NUMERIC	国际长途假日通话时长
	IDD_HL_CALL_CNT	NUMERIC	国际长途假日通话次数
	DDA_PK_CALL_LENGTH	NUMERIC	长途忙时通话时长
	DDA_PK_CALL_CNT	NUMERIC	长途忙时通话次数
	DDA_OP_CALL_LENGTH	NUMERIC	长途闲时通话时长
	DDA_OP_CALL_CNT	NUMERIC	长途闲时通话次数
	DDA_WK_CALL_LENGTH	NUMERIC	长途平时通话时长
	DDA_WK_CALL_CNT	NUMERIC	长途平时通话次数
	DDA_HL_CALL_LENGTH	NUMERIC	长途假日通话时长
	DDA_HL_CALL_CNT	NUMERIC	长途假日通话次数
	DDA_CALL_LENGTH	NUMERIC	长途通话时长
	DDA_CALL_CNT	NUMERIC	长途通话次数
	DDA_CALL_CHG	NUMERIC	长途通话费用
	RM_CALL_LENGTH	NUMERIC	国内漫游通话时长
	RM_CALL_CNT	NUMERIC	国内漫游通话次数
	RM_CALL_CHG	NUMERIC	国内漫游通话费用
	IRM_CALL_LENGTH	NUMERIC	国际漫游通话时长
	IRM_CALL_CNT	NUMERIC	国际漫游通话次数
	IRM_CALL_CHG	NUMERIC	国际漫游通话费用
	RMA_CALL_LENGTH	NUMERIC	漫游通话时长
	RMA_CALL_CNT	NUMERIC	漫游通话次数
	RMA_CALL_CHG	NUMERIC	漫游通话费用
	MBX_CNT	NUMERIC	语音信箱/移动秘书次数
	VAS_CNT	NUMERIC	增值服务次数
	VAS_CHG	NUMERIC	增值服务费用
	HVAS_CNT	NUMERIC	高额增值服务次数
	HVAS_CHG	NUMERIC	高额增值服务费用
	SMS_CNT	NUMERIC	SMS 次数
	SMS_CHG	NUMERIC	SMS 费用
	SMS_CP_CNT	NUMERIC	发向竞争对手的 SMS 次数
	SP_CNT	NUMERIC	SP 次数
	SP_CHG	NUMERIC	SP 费用
	EBIZ_CNT	NUMERIC	电子商务次数
	EBIZ_CHG	NUMERIC	电子商务费用
	VH_CALL_CNT	NUMERIC	高额呼叫次数
	VL_CALL_CNT	NUMERIC	特长呼叫次数
	L_CALL_CNT	NUMERIC	长时呼叫次数
	M_CALL_CNT	NUMERIC	一般呼叫次数
	S_CALL_CNT	NUMERIC	短时呼叫次数

续表

类别	列 名 称	列 类 型	列 描 述
	VS_CALL_CNT	NUMERIC	超短话单次数
	VL_FREE_CNT	NUMERIC	特长免费呼叫次数
	L_FREE_CNT	NUMERIC	长时免费呼叫次数
	M_FREE_CNT	NUMERIC	一般免费呼叫次数
	S_FREECNT	NUMERIC	短时免费呼叫次数
	VS_FREE_CNT	NUMERIC	超短免费话单次数
	LOC_VH_CALL_CNT	NUMERIC	区内高额呼叫次数
	LOC_VL_CALL_CNT	NUMERIC	区内特长呼叫次数
	LOC_L_CALL_CNT	NUMERIC	区内长时呼叫次数
	LOC_M_CALL_CNT	NUMERIC	区内一般呼叫次数
	LOC_S_CALL_CNT	NUMERIC	区内短时呼叫次数
	LOC_VS_CALL_CNT	NUMERIC	区内超短话单次数
	DDA_VH_CALL_CNT	NUMERIC	长途高额呼叫次数
	DDA_VL_CALL_CNT	NUMERIC	长途特长呼叫次数
	DDA_L_CALL_CNT	NUMERIC	长途长时呼叫次数
	DDA_M_CALL_CNT	NUMERIC	长途一般呼叫次数
	DDA_S_CALL_CNT	NUMERIC	长途短时呼叫次数
	DDA_VS_CALL_CNT	NUMERIC	长途超短话单次数
	RMA_VH_CALL_CNT	NUMERIC	漫游高额呼叫次数
	RMA_VL_CALL_CNT	NUMERIC	漫游特长呼叫次数
	RMA_L_CALL_CNT	NUMERIC	漫游长时呼叫次数
行为信息	RMA_M_CALL_CNT	NUMERIC	漫游一般呼叫次数
	RMA_S_CALL_CNT	NUMERIC	漫游短时呼叫次数
	RMA_VS_CALL_CNT	NUMERIC	漫游超短话单次数
	PK_VH_CALL_CNT	NUMERIC	忙时高额呼叫次数
	PK_VL_CALL_CNT	NUMERIC	忙时特长呼叫次数
	PK_L_CALL_CNT	NUMERIC	忙时长时呼叫次数
	PK_M_CALL_CNT	NUMERIC	忙时一般呼叫次数
	PK_S_CALL_CNT	NUMERIC	忙时短时呼叫次数
	PK_VS_CALL_CNT	NUMERIC	忙时超短话单次数
	OP_VH_CALL_CNT	NUMERIC	闲时高额呼叫次数
	OP_VL_CALL_CNT	NUMERIC	闲时特长呼叫次数
	OP_L_CALL_CNT	NUMERIC	闲时长时呼叫次数
	OP_M_CALL_CNT	NUMERIC	闲时一般呼叫次数
	OP_S_CALL_CNT	NUMERIC	闲时短时呼叫次数
	OP_VS_CALL_CNT	NUMERIC	闲时超短话单次数
	HL_VH_CALL_CNT	NUMERIC	假日高额呼叫次数
	HL_VL_CALL_CNT	NUMERIC	假日特长呼叫次数
	HL_L_CALL_CNT	NUMERIC	假日长时呼叫次数
	HL_M_CALL_CNT	NUMERIC	假日一般呼叫次数
	HL_S_CALL_CNT	NUMERIC	假日短时呼叫次数
	HL_VS_CALL_CNT	NUMERIC	假日超短话单次数
	IP_CALL_LENGTH	NUMERIC	IP 长途通话时长
	IP_CALL_CNT	NUMERIC	IP 长途通话次数

续表

类别	列　名　称	列　类　型	列　描　述
行为信息	IP_CALL_CHG	NUMERIC	IP 长途通话费用
	IP_DD_CALL_LENGTH	NUMERIC	IP 国内长途通话时长
	IP_DD_CALL_CNT	NUMERIC	IP 国内长途通话次数
	IP_DD_CALL_CHG	NUMERIC	IP 国内长途通话费用
	IP_IDD_CALL_LENGTH	NUMERIC	IP 国际长途通话时长
	IP_IDD_CALL_CNT	NUMERIC	IP 国际长途通话次数
	IP_IDD_CALL_CHG	NUMERIC	IP 国际长途通话费用
	FW_LENGTH	NUMERIC	呼转到固话通话时长
	FW_CNT	NUMERIC	呼转到固话通话次数
	FW_CHG	NUMERIC	呼转到固话通话费用
	XFW_LENGTH	NUMERIC	呼转到小灵通通话时长
	XFW_CNT	NUMERIC	呼转到小灵通通话次数
	XFW_CHG	NUMERIC	呼转到小灵通通话费用
	CFW_LENGTH	NUMERIC	呼转到竞争对手号码通话时长
	CFW_CNT	NUMERIC	呼转到竞争对手号码通话次数
	CFW_CHG	NUMERIC	呼转到竞争对手号码通话费用
	FWA_LENGTH	NUMERIC	呼转通话时长
	FWA_CNT	NUMERIC	呼转通话次数
	FWA_CHG	NUMERIC	呼转通话费用
	CC_LENGTH	NUMERIC	呼叫竞争对手客服号通话时长
	CC_CNT	NUMERIC	呼叫竞争对手客服号通话次数
	AVG_SLP_LENGTH	NUMERIC	平均睡眠时间
	MAX_SLP_LENGTH	NUMERIC	最长睡眠时间
	AVG_DDA_SLP_LENGTH	NUMERIC	长途平均睡眠时间
	AVG_DDA_SLP_LENGTH	NUMERIC	长途最长睡眠时间
	AVG_LOC_SLP_LENGTH	NUMERIC	区内通话平均睡眠时间
	AVG_LOC_SLP_LENGTH	NUMERIC	区内通话最长睡眠时间
	AVG_SMS_SLP_LENGTH	NUMERIC	SMS 平均睡眠时间
	AVG_SMS_SLP_LENGTH	NUMERIC	SMS 最长睡眠时间

10.2.3　建模过程

重入网识别主要包括基于 IMEI 串号和基于呼叫指纹两种方法。IMEI（International Mobile Equipment Identity，国际移动设备辨识码）是由 15 位数字组成的“电子串号”，与每台手机一一对应，而且是全球唯一的。每一部手机在组装完毕后都被赋予一个全球唯一的一组号码，这个号码从生产到交付使用都将被生产制造的厂商所记录。但是，由于种种原因目前大多数 CDMA 用户（WAP 用户除外）无法获取 IMEI 串号。此外，一些水货手机的

IMEI 串号无效等原因,无法完全通过 IMEI 串号识别重入网。通常情况下,基于 IMEI 串号和基于呼叫指纹两种方法联合使用,以确保重入网识别的准确率。所谓"呼叫指纹"是指用户在使用运营商的产品和服务过程中所产生的交往圈、呼叫特征、短信特征、位置特征、客服特征和终端特征等。由于这些信息对于一个用户而言是相对稳定的,且不同用户之间具有较大差异,因此可用于识别重入网。基于呼叫指纹的重入网识别是根据历史用户已经发生过的通话行为,经过科学分析,从通话呼叫行为中发现用户通话习惯、通话行为和通话交际圈等特征,利用这些显著特征标识用户,建立用户的呼叫行为档案,如同用户的"指纹"一样,每个用户的呼叫行为都不尽相同。通过广泛的用户呼叫行为分析,建立呼叫指纹库。在识别重入网时,首先在新发展用户群中过滤出疑似重入网用户,然后对疑似用户进行呼叫行为分析,同样可以获得用户呼叫行为特征,将新的呼叫行为特征纳入已经建立的呼叫指纹库,通过对比新旧呼叫指纹的相似度,判断疑似用户是否为重入网用户。

基于呼叫指纹的重入网识别的建模过程如下:

1. 确定待匹配用户和新入网用户清单

呼叫指纹识别需要建立新入网用户群和待匹配用户群两个数据集。

2. 选择特征变量和数据清洗

首先筛选合适的特征变量,表 10.10 列出了部分关键指标。

表 10.10 重入网识别的特征变量

序号	指 标 名 称
1	前 10 个最频繁通话号码重合率
2	前 10 个最长的总通话时长号码重合率
3	前 10 个最长的单次通话时长号码重合率
4	前 10 个总通话次数最多号码重合率
5	前 10 个点对点发送短信最频繁号码的重合率
6	点对点短信次数的变异率
7	通话小区数的变异率
8	本网交际圈人数的重合率
9	他网交际圈人数的变异率
10	前 10 个最频繁通话时段的重合率
11	…

3. 建立呼叫指纹库

建立呼叫指纹库的关键是筛选出可以辨别不同用户身份的特征号码库,并逐一计算其权重,即建立特征号码权重库。首先,统计用户拨打的所有特征电话号码的频次,从高到低排序,剔除拨打频次最高的一部分公共号码以及拨打频次最低的一部分稀疏号码,这些号码对于区分用户的呼叫指纹没有实际意义;然后,对剩余的特征号码赋予不同的权重,建立特征号码权重库。特征号码权重的计算采用 TFIDF 算法。

重入网识别引入 TFIDF 算法的优势在于随着用户拨打特征号码 t_i 次数的增加,即随着用户拨打相同的特征号码次数的增加,则 $\mathrm{tf}(t_i,d)$的值增加,该特征号码的权重将增加,

这与我们通常的理解一致；如果该特征号码被许多用户所拨打，则 idf(t_i)的值反而减小，特征号码的权重将减小，表明该特征号码对辨别用户身份的作用将减小，这是因为如果该特征号码也被其他用户频繁地拨打，如公共号码，则其分辨个体特征的能力相对减弱，所以其权重应当减小，以便使那些更能分辨用户是否具有相似呼叫指纹的特征号码的权重相应增大。

采用 TFIDF 算法在一定程度上减少了“噪声”号码对判别准确度的影响，突出重要的特征号码，同时又考虑了整个呼叫指纹库呼叫特征之间的关系。因此如果样本量足够，最终重入网识别具有较高的准确率。

4. 计算呼叫指纹相似度

呼叫指纹相似度是指新入网用户与待匹配用户之间的呼叫行为特征相似程度，即每个特征变量的重合率或变异率的加权值，其计算公式为：

$$S = a_1 \times x_1 + a_2 \times x_2 + \cdots + a_n \times x_n \qquad (10.6)$$

其中，x_i 代表新入网用户与待匹配用户在第 i 个特征变量的重合率或变异率，a_i 代表每个特征变量的权重，根据其在重入网识别中的重要程度可以动态调整。

5. 设定呼叫指纹相似度阈值

大于该阈值的匹配用户对可界定为疑似重入网用户。

6. 验证

通过电话外呼或客户资料比照可以对疑似重入网用户进行验证以锁定重入网用户，检验模型的准确性及覆盖率是否达到预期要求，如果达不到要求则需要调整模型。

锁定重入网用户后，需要进一步将重入网用户与订购套餐/产品、渠道等进行关联分析，发现引起重入网的症结所在，推荐合适的套餐，降低离网率和欠费风险，降低经营风险。

基于 IMEI 串号的重入网识别需要进行数据清洗以排除噪声，即排除错误的、非法烧录的、空缺的 IMEI 串号等，然后对 IMEI 串号进行逐一扫描和匹配，定位重入网用户。

10.2.4 结果

利用基于 IMEI 串号和基于呼叫指纹两种重入网识别方法，对某市 2008 年 9 月流失的 CDMA 用户 45 537 户和 2008 年 9 月和 10 月两个月新发展的 CDMA 用户 135 638 户进行重入网甄别，通过匹配找到拨打过相同特征号码的用户对共计 193 281，其中大部分流失用户在新入网用户中都能找到多于 1 个的匹配用户，逐一计算相似度，设定相似度阈值为 50，找到疑似重入网用户 12 596，对这些疑似重入网用户进行客户资料比对和电话回访进行确认，最终锁定重入网用户 10 791，占 2008 年 9 月流失的 CDMA 用户的 23.7%。

重入网识别的实际意义和作用体现在如下两个方面。

1. 调整新入网政策

- 针对部分用户自身的流动性和不稳定性，对具有流动性的用户可以采取针对性的绑定措施进行挽留。
- 对新发展用户中不少用户入网定位不准，进行深度细分，并针对性地提前开展主动的市场梳理，减少重入网现象。

- 对重入网用户开展重入网原因和政策了解程度的调查,调整产品和套餐漏洞,减少成本和收入损失。

2. 调整代理商佣金政策

代理商佣金侧重奖励新发展的非重入网用户,适当降低新发展的重入网用户的佣金标准。

此外,通过在用户、业务、品牌、渠道、消费层次和生命周期等多个方面对重入网现象进行分析,找出不足并采取措施。

10.3 虚开欺诈识别

10.3.1 定义

虚开主要是指客户资料虚假、开户之后没有或者几乎没有有效的话单,但运营商却要支付一定比例的代理费。虚开用户带给运营商的收入很少或者几乎没有,反而增加了成本,造成不少损失。

虚开用户主要表现在如下几个方面。

1. 很少的费用

虚开用户出账费用一般很少,小于新入网用户平均出账费用的一半以上,大部分虚开用户根本不出账。

2. 很少的话务

如果开户后一月之内没有或者很少的有效话单,这些用户很可能是虚开用户;如果通话次数或者通话时长低于某一阈值,则也可能是虚开用户。

3. 很少的缴费

虚开用户很少去缴费,或者缴费金额很少。

4. 用户资料虚假

多数虚开用户的资料是虚假的,如没有姓名或者身份证号码;格式不正确或者身份证号码有误;甚至没有提供用户资料。

10.3.2 数据准备

根据业务需求选取某市2008年12月新发展的146 283个CDMA用户作为分析对象,从ODS和业务支撑系统,抽取有效的新发展用户连续三个月的通话详单和出账账单数据,合并汇总后生成数据宽表。

通过数据清洗剔除下列无效用户,即:

- CDMA智能网用户和C+W用户
- 当月离网的新发展用户
- 找不到资料的用户

清洗后当月有效的新发展用户数为125 629,其中CDMA准预付费用户数为39 107,CDMA后付费用户数为86 522。

10.3.3 建模过程

虚开欺诈识别的建模过程如下：

(1) 通话行为筛选

为了缩小虚开欺诈的搜索范围，对于当月新发展用户，首先根据当月总通话次数和时长等指标是否大于当月在网用户的平均值进行排除，小于的为疑似虚开用户。

(2) 消费行为筛选

接下来对疑似虚开用户，根据其日均消费是否大于在网用户日均消费再次进行排除，小于的为疑似虚开用户。

(3) 通话间隔筛选

针对上述两轮排查均为疑似虚开的用户，利用其入网以来，前 100 次通话的时间间隔进行聚类，最终锁定虚开用户。

10.3.4 结果

按照上述分析过程，对于当月有效的 125 629 新发展用户，经过通话行为筛选，发现疑似虚开用户 65 318；再经过消费行为筛选，发现疑似虚开用户 49 423；最后，对 49 423 疑似虚开用户的前 100 次通话的时间间隔进行聚类，由于篇幅所限，这里仅给出了前 30 次通话的时间间隔(以小时为单位)的聚类结果，如表 10.11 所示。

表 10.11 前 30 个通话的时间间隔聚类结果

	类别				
	1	2	3	4	5
通话时间间隔 1	26.72	1158.12	154.75	13.08	142.73
通话时间间隔 2	10.48	1419.02	34.37	6.31	50.75
通话时间间隔 3	10.60	1487.73	17.78	23.91	35.13
通话时间间隔 4	12.65	无通话	13.96	214.10	30.95
通话时间间隔 5	18.41	无通话	13.10	640.00	39.83
通话时间间隔 6	15.85	无通话	11.86	915.32	29.92
通话时间间隔 7	94.70	无通话	11.88	1223.89	41.48
通话时间间隔 8	80.80	无通话	10.83	无通话	44.99
通话时间间隔 9	633.97	无通话	10.52	无通话	34.58
通话时间间隔 10	1012.82	无通话	9.98	无通话	33.24
通话时间间隔 11	1215.38	无通话	9.44	无通话	26.61
通话时间间隔 12	1327.17	无通话	10.56	无通话	24.33
通话时间间隔 13	1401.27	无通话	9.74	无通话	24.28
通话时间间隔 14	1456.44	无通话	9.08	无通话	27.11
通话时间间隔 15	无通话	无通话	9.27	无通话	95.55

续表

	类　别				
	1	2	3	4	5
通话时间间隔 16	无通话	无通话	9.00	无通话	255.44
通话时间间隔 17	无通话	无通话	12.80	无通话	398.88
通话时间间隔 18	无通话	无通话	8.85	无通话	520.89
通话时间间隔 19	无通话	无通话	9.01	无通话	664.66
通话时间间隔 20	无通话	无通话	8.78	无通话	800.68
通话时间间隔 21	无通话	无通话	8.25	无通话	913.36
通话时间间隔 22	无通话	无通话	9.03	无通话	1020.93
通话时间间隔 23	无通话	无通话	8.15	无通话	1126.56
通话时间间隔 24	无通话	无通话	8.56	无通话	1235.35
通话时间间隔 25	无通话	无通话	7.94	无通话	1346.50
通话时间间隔 26	无通话	无通话	8.75	无通话	1468.29
通话时间间隔 27	无通话	无通话	16.74	无通话	无通话
通话时间间隔 28	无通话	无通话	29.68	无通话	无通话
通话时间间隔 29	无通话	无通话	37.89	无通话	无通话
通话时间间隔 30	无通话	无通话	48.63	无通话	无通话

类别	1	3205
	2	4782
	3	36 584
	4	3491
	5	1361
Valid		49423
Missing		0

由表 10.11 可知：

第一类用户 3205 户，入网一天内发生了通话行为，表面上看这类用户不是虚开用户。三天内通话 5 次，但第 7 和 8 次通话间隔持续一周左右。

第二类用户 4782 户，入网后基本没有通话。

第三类用户 36 584 户，通话时间间隔较为均匀，为正常使用的用户。

第四类用户 3491 户，入网后两天内通话 3 次，第 3 和第 4 次通话的时间间隔近 10 天，第 5 和第 6 次通话的时间间隔更长达 2 个月。

第五类用户 1361 户，入网初期通话行为与正常用户相似，只是通话间隔是正常用户群的 3～4 倍，但随着时间的推移，通话逐渐稀少，这类用户存在离网风险。

最终锁定第一、二和四类为虚开欺诈用户共计 11 478，占当月有效新发展用户 125 629 的 9.1%。

对虚开欺诈用户首次拨打的电话号码排序，排名前 10 的电话号码如表 10.12 所示。

表 10.12　虚开欺诈用户首次拨打的电话号码排名(前 10 个)

电 话 号 码	呼 叫 次 数	说　　明
969001	7436	就业招聘
16882777	2964	收费节目
95555	2618	招商银行服务
11888	1287	电信充值
969221	950	天气预报
12121	814	天气预报
16882660	480	
16885000	423	
16882659	389	

具体地,对首次拨打号码 95555 的虚开用户的通话行为进行跟踪,发现具有一定规律性,如表 10.13 所示。

表 10.13　首次拨打号码为 95555 的虚开用户实例

入 网 时 间	号　　码	呼叫号码 1	呼叫号码 2	呼叫号码 3	呼叫号码 4	呼叫号码 5	呼叫号码 6	呼叫号码 7	呼叫号码 8	呼叫号码 9	呼叫号码 10
2008-12-15 16:36	153××××5798	95555	95555	95555	95555	95555	95555	95555	11888	再无通话	再无通话
2008-12-15 16:33	153××××5838	95555	95555	95555	95555	95555	95555	95555	11888	再无通话	再无通话
2008-12-15 16:31	153××××5878	95555	95555	95555	95555	95555	95555	95555	11888	11888	再无通话
2008-12-15 16:28	153××××5908	95555	95555	95555	95555	95555	95555	95555	11888	11888	再无通话
2008-12-15 11:04	153××××5503	95555	95555	95555	95555	95555	95555	95555	11888	再无通话	再无通话
2008-12-15 11:07	153××××5506	95555	95555	95555	95555	95555	95555	95555	11888	再无通话	再无通话
2008-12-15 12:25	153××××5512	95555	95555	95555	95555	95555	95555	95555	11888	再无通话	再无通话
2008-12-15 12:25	153××××5513	95555	95555	95555	95555	95555	95555	95555	11888	再无通话	再无通话
2008-12-1512:35	153××××5515	95555	95555	95555	95555	95555	95555	95555	11888	11888	再无通话
2008-12-15 12:45	153××××5520	95555	95555	95555	95555	95555	95555	95555	11888	再无通话	再无通话

此外,入网后首次拨打号码 969001 和 16882777 的虚开用户也存在类似现象。

虚开用户数量与入网时间的关系如图 10.6 所示。

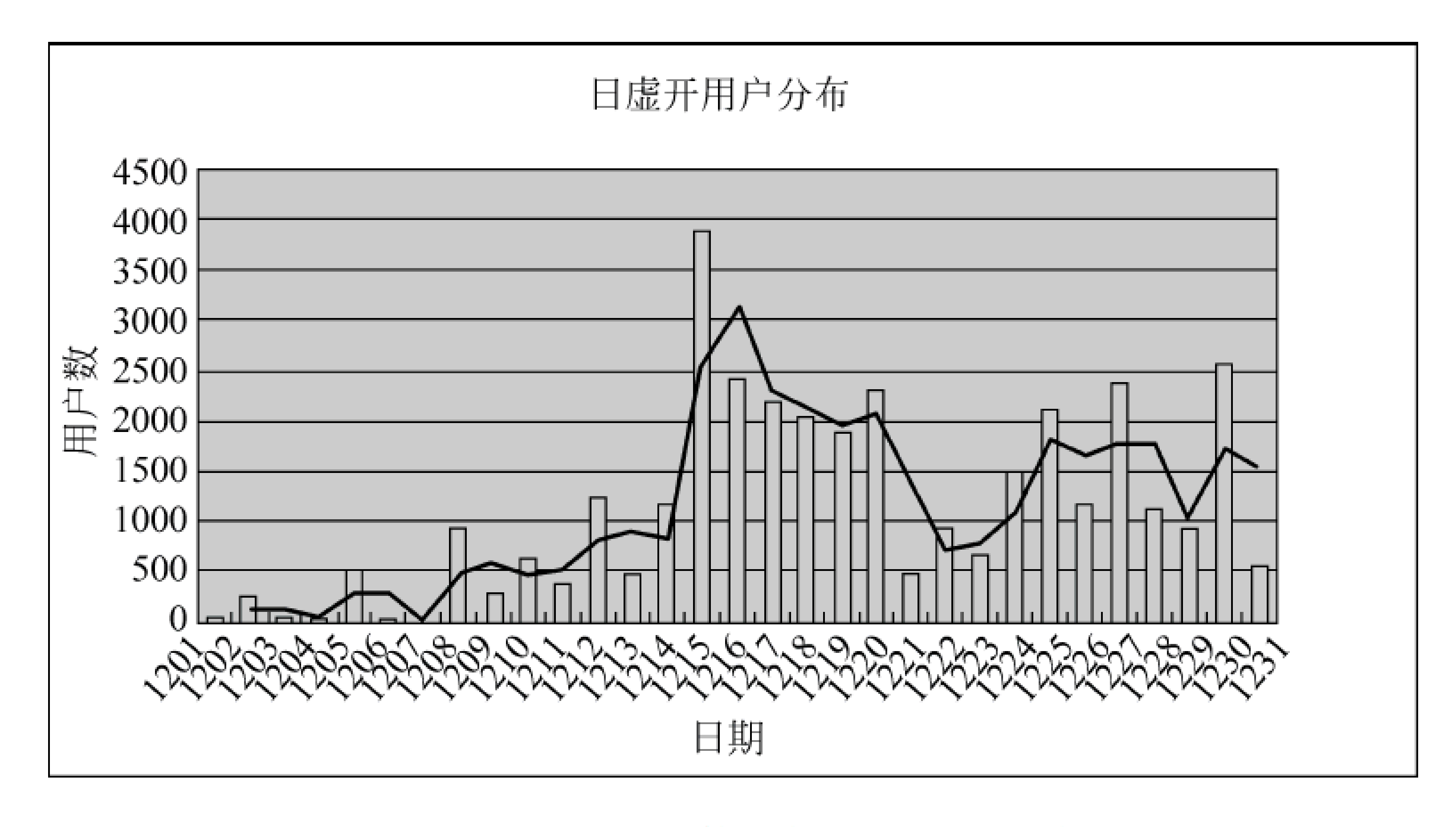


图 10.6　虚开用户数量与虚开时间的关系

由图 10.6 可知,虚开用户多发生在月中至月底,当月 15 日产生峰值。

经过分析,虚开行为的动机主要可归纳为:

- 销售政策不合理,造成代理商骗取佣金、套机套卡等。
- 政策指标压力大,为了完成任务,拼命发展用户,不顾质量或者弄虚作假。
- 用户满意度不高,对资费、服务、网络等方面不满,初期使用了一两次后就再也没有兴趣使用。
- 其他因素。SP 自消费、偶然因素、用户自身短期流动和买卡后即离开等。

通过虚开欺诈识别可以对新发展用户质量进行监控,规避市场风险。目前国内各电信运营商的移动业务发展任务指标都很重。在保证发展数量的同时,更应该注重发展质量,加强渠道管控。

10.4 数据业务收入预测

10.4.1 定义

目前,数据业务逐步成为电信业务收入新的增长点,3G 时代数据业务收入的保障对于电信运营商未来发展具有极其重要的战略意义。数据业务的发展空间较大,同时竞争也日趋加剧,因此及时掌握数据业务收入情况对发展尤为重要。

数据业务收入预测是指逐月滚动预测未来收入情况,根据过去及本年度各月收入情况,预测本年度各月的收入,提前预知各月数据业务收入情况,以便于及时了解收入异动原因,采取相应措施。

10.4.2 数据准备

数据业务收入预测的数据来源于某移动公司的经营分析系统(以下简称 BI),其中包括客户信息资料、消费账单和签约信息等。这些数据都存储在经分数据仓库中(IBM DB2),客户资料来源于 OCRM 系统,包括客户的基本信息如客户名称、签约时间等;客户账单来源于 BOSS 系统,分为实时账单、日账单和月账单,包括客户所用移动产品的具体付费情况。BOSS 和 OCRM 接口传来的数据具有量大、属性多、定义复杂和不完整等特性,经过 BI 系统的处理,经过一套数据转换、清洗、汇总等流程转化为干净可用的数据,具体的数据加工过程如图 10.7 所示。

数据业务收入预测的输入均为经过 BI 加工处理后的数据,是数据仓库中的基础表,不仅为数据业务收入预测服务,而且还为 BI 其他应用服务,因此这些数据表均可以直接应用,这些表主要包含 BOSS 系统的各类计费详单、OCRM 系统的用户资料等,具体如表 10.14 所示。

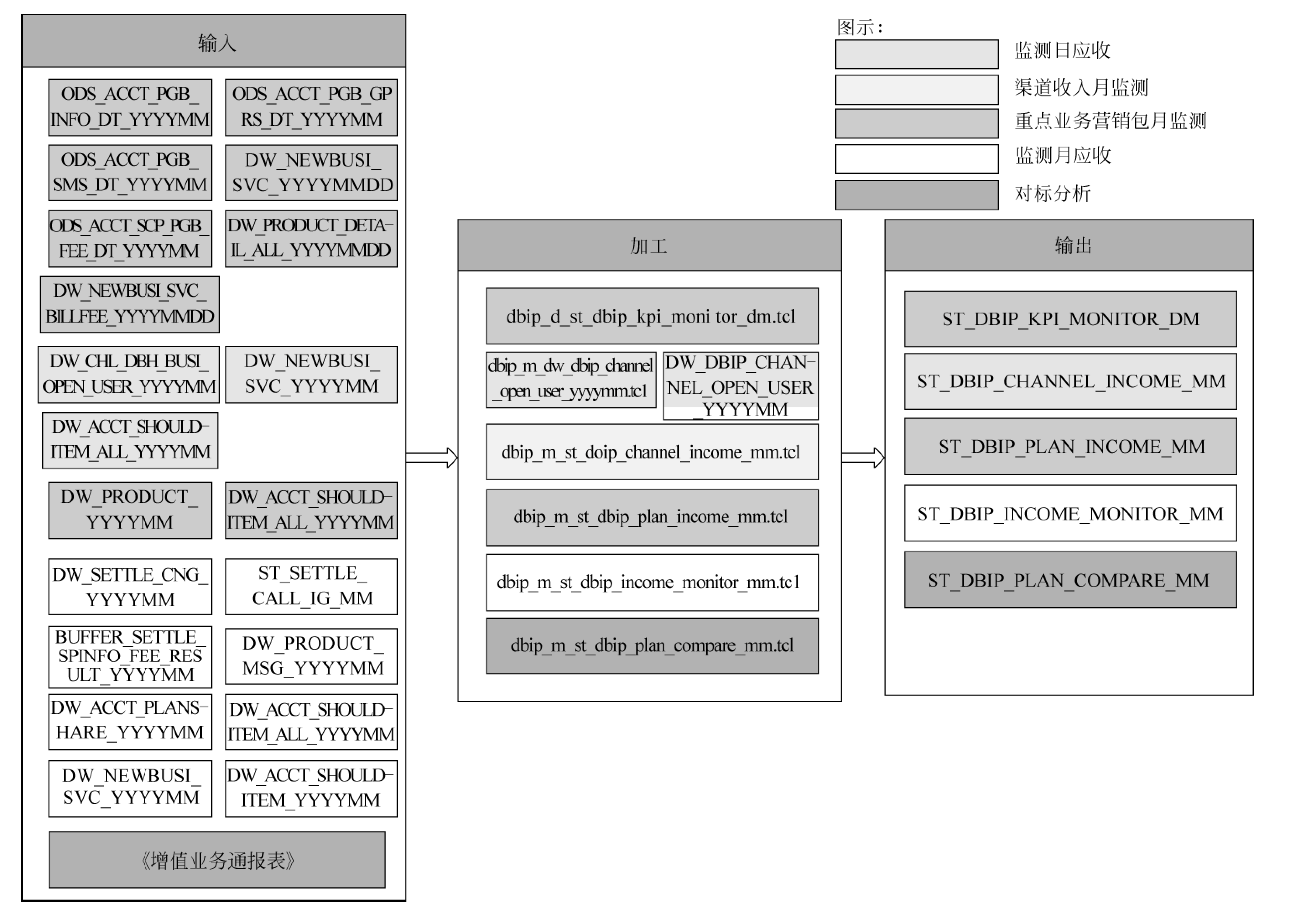


图 10.7　数据加工处理过程的示意图

表 10.14　数据业务收入的输入表

表　名　称	说　　明
ODS_ACCT_PGB_INFO_DT_YYYYMM	信息费计费详单月累积表
ODS_ACCT_PGB_GPRS_DT_YYYYMM	GRPS 计费详单月累积表
ODS_ACCT_PGB_SMS_DT_YYYYMM	SMS 计费详单月累积表
YQ_NEWBUSI_SVC_YYYYMMDD	数据业务使用情况日表
ODS_ACCT_SCP_PGB_FEE_DT_YYYYMM	智能网 PGB 费用月累积表
YQ_PRODUCT_DETAIL_ALL_YYYYMMDD	用户资料日表
YQ_NEWBUSI_SVC_BILLFEE_YYYYMMDD	数据业务费用日表
YQ_CHL_DBH_BUSI_OPEN_USER_YYYYMM	渠道用户业务开通月表
YQ_NEWBUSI_SVC_YYYYMM	数据业务使用情况月表
YQ_ACCT_SHOULDITEM_ALL_YYYYMM	用户费用账单月表
YQ_PRODUCT_YYYYMM	用户资料月表
YQ_SETTLE_CNG_YYYYMM	CNG 结算费用月表
ST_SETTLE_CALL_IG_MM	IG 结费费用月表
BUFFER_SETTLE_SPINFO_FEE_RESULT_YYYYMM	SP 信息费结算月表
YQ_PRODUCT_MSG_YYYYMM	用户资料信息月表
YQ_ACCT_PLANSHARE_YYYYMM	用户分摊费用月表
YQ_ACCT_SHOULDITEM_YYYYMM	非智能网用户账单月表

为了满足数据业务收入预测的需求，需要将所依赖的各输入数据源进行加工处理，转换成易于前端展示的更粗粒度的汇总数据，具体数据处理程序如表 10.15 所示。

表 10.15　数据业务收入的数据处理程序列表

程序名称	说　　明
dbip_d_st_dbip_kpi_monitor_dm.tcl	收入日监控数据处理程序
dbip_m_dw_dbip_channel_open_user_yyyymm.tcl	渠道收入月监控仓库层数据处理程序
dbip_m_st_dbip_channel_income_mm.tcl	渠道收入月监控应用层数据处理程序
dbip_m_st_dbip_plan_income_mm.tcl	重点业务营销包月数据处理程序
dbip_m_st_dbip_income_monitor_mm.tcl	收入月监控数据处理程序
dbip_m_st_dbip_plan_compare_mm.tcl	对标分析数据处理程序

以收入日监控数据处理程序为例，需要对每日数据业务的总收入及点对点短信、彩信、WAP 和飞信的收入进行按日汇总，生成日 ST 表，具体的加工处理过程如表 10.16 所示。

表 10.16　收入日监测程序的处理过程

项　　目	说　　明
模块程序名	dbip_d_st_dbip_kpi_monitor_dm.tcl
输入、输出参数	输入参数：op_time(数据日期)
输入数据源	YQ_NEWBUSI_SVC_BILLFEE_YYYYMMDD YQ_NEWBUSI_SVC_YYYYMMDD ODS_ACCT_SCP_PGB_FEE_DT_YYYYMM ODS_ACCT_PGB_SMS_DT_YYYYMM ODS_ACCT_PGB_GPRS_DT_YYYYMM ODS_ACCT_PGB_INFO_DT_YYYYMM YQ_PRODUCT_DETAIL_ALL_YYYYMMDD

续表

<table>
<tr><th>项　　目</th><th>说　　明</th></tr>
<tr><td>输出结果</td><td>ST_DBIP_KPI_MONITOR_DM</td></tr>
<tr><td>前置条件</td><td>1. 每日所依赖的源系统接口传输完成
2. BI 数据仓库汇总完成</td></tr>
<tr><td>处理逻辑</td><td>输　入
ODS_ACCT_PGB_INFO_DT_YYYYMM
ODS_ACCT_PGB_GPRS_DT_YYYYMM
ODS_ACCT_PGB_SMS_DT_YYYYMM
DW_NEWBUSI_SVC_YYYYMMDD
ODS_ACCT_SCP_PGB_FEE_DT_YYYYMM
DW_PRODUCT_DETAIL_ALL_YYYYMMDD
DW_NEWBUSI_SVC_BILLFEE_YYYYMMDD
加　工
dbip_d_st_dbip_kpi_monitor_dm.tcl
输　出
ST_DBIP_KPI_MONITOR_DM</td></tr>
<tr><td>统计口径</td><td>1. 数据业务总收入
数据业务总收入＝总收入－非数据业务的语音及语音增值业务费
其中：
总收入＝四类详单(语音详单、INFO 详单、GPRS 详单、SMS 详单)合计＋动感地带月租费＋智能网月租费
统计脚本(SQL)：
(1) 非智能网收入-新业务费

select
   Sum(value(a.sms_basefee,0)) + sum(value(a.info_fee,0) + value(a.monthinfo_fee,0)) + sum(value(a.flow_fee,0))
   as kpi_value
from
   dw_newbusi_svc_billfee_${yyyymmdd} a
left outer join
   dw_product_detail_all_${yyyymmdd} b
   on a.product_no = b.product_no
where
   (b.brand_id not in (6,7,8)
   or b.sub_brand_id in(6110,7107,6414,6616,6131,6132,6133,6134)
   or b.brand_id is null)
   and a.srctype_id < 200

(2) 智能网收入-PGB

select
   sum(info_fee * 0.01) as kpi_value
from
   DWRUN.ODS_ACCT_SCP_PGB_FEE_DT_${yyyymm} a
left outer join
   dw_product_detail_all_${yyyymmdd} b
   on a.msisdn = b.product_no
where
   a.payfee_status_id = '000'
   and a.op_time = '${yyyy-mm-dd}'</td></tr>
</table>

续表

项　　目	说　　明
统计口径	(3) 智能网收入-新业务费 select 　　Sum(value(sms_basefee,0)) + sum(value(info_fee,0) 　　+ value(monthinfo_fee,0)) + sum(value(flow_fee,0)) 　　as kpi_value From 　　dw_newbusi_svc_billfee_${yyyymmdd} a 　　left outer join dw_product_detail_all_${yyyymmdd} b 　　on a.product_no = b.product_no Where 　　b.brand_id in (6,7,8) 　　And b.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134) 　　And a.srctype_id>=200 2. 点对点短信收入 点对点短信 ＝ 网内点对点短信 ＋ 企信通网内短信费 ＋ 网间点对点短信 ＋ 校信通业务费 ＋ 企信通联通短信费 ＋ 本地国际短信费 统计脚本(SQL)： (1) 非智能网-点对点短信收入 Select 　　sum(value(basecall_fee,0) + value(toll_fee1,0) 　　+ value(toll_fee2,0) + value(toll_fee3,0)) from dw_newbusi_svc_${yyyymmdd} a left outer join 　　dw_product_detail_all_${yyyymmdd} c 　　on a.product_no = c.product_no left outer join 　　DIM_ACCT_CHARGECODE b 　　on a.basecall_fee_id = b.charge_code where 　　basecall_fee_id in ('50','306','307','51','112') 　　and (not (c.brand_id in (6,7,8) 　　and c.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134)) 　　or c.brand_id is null) 　　And srctype_id<200 (2) 智能网-点对点短信收入 Select 　　sum(value(basecall_fee,0) + value(toll_fee1,0) 　　+ value(toll_fee2,0) + value(toll_fee3,0)) from 　　dw_newbusi_svc_${yyyymmdd} a left outer join 　　dw_product_detail_all_${yyyymmdd} c 　　on a.product_no = c.product_no left outer join

续表

项　　目	说　　明
统计口径	DIM_ACCT_CHARGECODE b on a.basecall_fee_id = b.charge_code where basecall_fee_id in ('50','306','307','51','112') and c.brand_id in (6,7,8) and c.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134) And srctype_id >= 200 (3) 智能网-本地国际短信费 select sum(value(basecall_fee,0) + value(toll_fee1,0) + value(toll_fee2,0) + value(toll_fee3,0)) from dw_newbusi_svc_ $ {yyyymmdd} a left outer join dw_product_detail_all_ $ {yyyymmdd} c on a.product_no = c.product_no left outer join DIM_ACCT_CHARGECODE b on a.basecall_fee_id = b.charge_code where svcitem_id = 'BI180200' and basecall_fee_id = '0' and c.brand_id in (6,7,8) and c.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134) And srctype_id >= 200 (4) 智能网-点对点短信收入-PGB select sum(value(AMOUNT_1,0) + value(AMOUNT_2,0) + value(AMOUNT_3,0) + value(AMOUNT_4,0)) as kpi_value from DWRUN.ODS_ACCT_PGB_SMS_DT_ $ {yyyymm} a left outer join dw_product_detail_all_ $ {yyyymmdd} b on a.product_no = b.product_no where a.op_time = '$ {yyyy-mm-dd}' and charge_code_1 in ('50','306','307','51','112') (5) 企信通网内短信费 select sum(value(basecall_fee,0) + value(toll_fee1,0) + value(toll_fee2,0) + value(toll_fee3,0)) as kpi_value from dw_newbusi_svc_2001114 a left outer join

续表

项　　目	说　　明
统计口径	DIM_ACCT_CHARGECODE b on a.basecall_fee_id = b.charge_code left outer join dw_product_detail_all_ $ {yyyymmdd} c on a.product_no = c.product_no where basecall_fee_id in ('301') 3. WAP 收入 WAP 收入 ＝ WAP 通信费 ＋ GPRS 国内流量费(减去全球通 随 e 行) ＋ GPRS 国际下传费用 ＋ 3G GPRS 流量费 ＋ 手机电视 3G 流量费 统计脚本(SQL): (1) WAP-非智能网 select sum(basecall_fee + toll_fee1 + toll_fee2 + toll_fee3) as kpi_value from DW_NEWBUSI_svc_ $ {yyyymmdd} a left outer join dw_product_detail_all_ $ {yyyymmdd} c on a.product_no = c.product_no left outer join DIM_ACCT_CHARGECODE b on a.basecall_fee_id = b.charge_code where (basecall_fee_id in ('38','420','199','428') or (basecall_fee_id = '195' and c.sub_brand_id not in(5121, 5124))) and (not (c.brand_id in (6,7,8) and c.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134)) or c.brand_id is null) And srctype_id < 200 (2) WAP-智能网 select sum(basecall_fee + toll_fee1 + toll_fee2 + toll_fee3) as kpi_value from DW_NEWBUSI_svc_ $ {yyyymmdd} a left outer join dw_product_detail_all_ $ {yyyymmdd} c on a.product_no = c.product_no left outer join DIM_ACCT_CHARGECODE b on a.basecall_fee_id = b.charge_code where (basecall_fee_id in ('38','420','199','428') or (basecall_fee_id = '195' and c.sub_brand_id not in(5121, 5124))) and c.brand_id in (6,7,8) and c.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134) And srctype_id >= 200

续表

项　　目	说　　明
统计口径	(3) WAP-PGB select sum(value(AMOUNT_1,0) + value(AMOUNT_2,0) + value(AMOUNT_3,0) + value(AMOUNT_4,0)) as kpi_value from DWRUN.ODS_ACCT_PGB_GPRS_DT_${yyyymm} a left outer join dw_product_detail_all_${yyyymmdd} b on a.product_no = b.product_no where a.op_time = '${yyyy-mm-dd}' and (charge_code_1 in ('38','420','199','428') or (charge_code_1 = '195' and b.sub_brand_id not in(5121, 5124))) 4. 点对点彩信收入 点对点彩信收入 ＝ 彩信用户通信费 ＋ 联通彩信通信费 ＋ 国际彩信通信费 统计脚本(SQL): (1) 点对点彩信-非智能网 select sum(value(basecall_fee,0) + value(toll_fee1,0) + value(toll_fee2,0) + value(toll_fee3,0)) as kpi_value from DW_NEWBUSI_svc_${yyyymmdd} a left outer join dw_product_detail_all_${yyyymmdd} c on a.product_no = c.product_no left outer join DIM_ACCT_CHARGECODE b on a.basecall_fee_id = b.charge_code where basecall_fee_id in ('85','319','226') and (not (c.brand_id in (6,7,8) and c.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134)) or c.brand_id is null) And srctype_id < 200 (2) 点对点彩信 - 智能网 select sum(value(basecall_fee,0) + value(toll_fee1,0) + value(toll_fee2,0) + value(toll_fee3,0)) as kpi_value from DW_NEWBUSI_svc_${yyyymmdd} a left outer join dw_product_detail_all_${yyyymmdd} c on a.product_no = c.product_no left outer join DIM_ACCT_CHARGECODE b on a.basecall_fee_id = b.charge_code where basecall_fee_id in ('85', '319','226') and c.brand_id in (6,7,8) and c.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134) And srctype_id >= 200

续表

项　目	说　明
统计口径	(3) 点对点彩信-PGB select sum(value(AMOUNT_1,0) + value(AMOUNT_2,0) + value(AMOUNT_3,0) + value(AMOUNT_4,0)) as kpi_value from DWRUN.ODS_ACCT_PGB_SMS_DT_ ${yyyymm} a left outer join dw_product_detail_all_ ${yyyymmdd} b on a.product_no = b.product_no where a.op_time = '${yyyy-mm-dd}' and charge_code_1 in ('85', '319','226') 5. 飞信收入 飞信收入 = 飞信通话费 + 飞信短信费 + 飞信速配交友功能费 统计脚本(SQL): (1) 飞信 - 非智能网用户 Select sum(value(basecall_fee,0) + value(toll_fee1,0) + value(toll_fee2,0) + value(toll_fee3,0)) as kpi_value from dw_newbusi_svc_ ${yyyymmdd} a left outer join dw_product_detail_all_ ${yyyymmdd} b on a.product_no = b.product_no where basecall_fee_id in ('281','265','275') and (not (b.brand_id in (6,7,8) and b.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134)) or b.brand_id is null) And srctype_id < 200 (2) 飞信-智能网用户 select sum(basecall_fee + toll_fee1 + toll_fee2 + toll_fee3) from dw_newbusi_svc_ ${yyyymmdd} a left outer join dw_product_detail_all_ ${yyyymmdd} b on a.product_no = b.product_no where basecall_fee_id in ('281','265','275') and b.brand_id in (6,7,8) and b.sub_brand_id not in(6110,7107,6414,6616,6131,6132,6133,6134) And srctype_id >= 200

续表

项目	说明
统计口径	(3) 飞信 - PGB ```select sum(value(AMOUNT_1,0) + value(AMOUNT_2,0) + value(AMOUNT_3,0) + value(AMOUNT_4,0)) as kpi_value from DWRUN.ODS_ACCT_PGB_INFO_DT_${yyyymm} a left outer join dw_product_detail_all_${yyyymmdd} b on a.product_no = b.product_no where a.op_time = '${yyyy-mm-dd}' and charge_code_1 in ('281','265','275')```
程序位置	/dw_run/bi_dw/aiomni/odsst/bin/DBIP

经过各数据处理程序对输入数据进行相应的汇总处理，形成了数据业务收入的结果表，用来存储各个模块的各统计周期数据，统计周期主要有日和月，具体如表 10.17 所示。

表 10.17 数据业务收入的输出表

表名称	说明
ST_DBIP_KPI_MONITOR_DM	收入日监控数据存储表
ST_DBIP_CHANNEL_INCOME_MM	渠道收入月监控应用层数据存储表
ST_DBIP_PLAN_INCOME_MM	重点业务营销包月监控数据存储表
ST_DBIP_INCOME_MONITOR_MM	收入月监控数据存储表
YQ_DBIP_CHANNEL_OPEN_USER_YYYYMM	渠道收入月监控仓库层数据存储表
ST_DBIP_PLAN_COMPARE_MM	对标分析月分析数据存储表
ST_DBIP_INCOME_FORECAST_MM	数据业务总收入预测数据存储表

数据业务收入预测的输出表为新增仓库表，需要在数据仓库中建立这些表以存储各功能模块的统计数据，因此需要对各库表的结构进行设计，主要包括各表的字段定义，具体各表的定义如表 10.18～表 10.24 所示。

其中，ST_DBIP_KPI_MONITOR_DM 是收入日监控表，用来存储日收入五个指标的维度值及指标值，维度主要分为数据日期和品牌，具体表的定义如表 10.18 所示。

表 10.18 收入日监控表

字段名称	字段说明
OP_TIME	数据日期
BRAND_ID	品牌 ID
KPI_ID	KPI 编号
KPI_NAME	KPI 名称
KPI_VALUE	KPI 值

ST_DBIP_PLAN_INCOME_MM 是重点业务营销包收入月监控表,用来存储各类营销包在各个维度的收入、当月收入、历史收入和累计收入,维度主要分为数据日期、品牌和营销包分类,具体表的定义如表 10.19 所示。

表 10.19　重点业务营销包月监控表

字段名称	字段说明
OP_TIME	数据日期
CLASS_ID	营销包分类编码
SUB_CLASS_ID	营销包子类编码
ITEM_ID	营销包编码
BRAND_ID	品牌
AMOUNT	本月
LAST_MONTH_AMOUNT	上月
LAST_YEAR_AMOUNT	去年同期
ACCUMULATE_AMOUNT	本年累计
LAST_ACCUMULATE	去年同期累计
TOTAL_AMOUNT	当年所有收入

ST_DBIP_INCOME_MONITOR_MM 是收入月监控表,用来存储各业务在各个维度的收入、当月收入、历史收入及累计收入,维度主要分为数据日期、品牌、收入分类和市场分类等,具体表的定义如表 10.20 所示。

表 10.20　收入月监控表

字段名称	字段说明
OP_TIME	数据日期
CLASS_ID	收入分类编码
ITEM_ID	ITEM_ID
BRAND_ID	品牌＝DIM_DBIP_PRODUCT_BRAND. BRAND_ID
MARKET_CLASS_ID	市场分类
PROFFER_CLASS_ID	业务贡献度
SOURCE_CLASS_ID	收入来源方式
AMOUNT	当月收入
LAST_MONTH_AMOUNT	上月收入
LAST_YEAR_AMOUNT	去年同期收入
ACCUMULATE_AMOUNT	累计收入
LAST_ACCUMULATE	去年同期累计收入
TOTAL_AMOUNT	当月累计总收入

OW_DBIP_CHANNEL_OPEN_USER_YYYYMM 是渠道收入月监控仓库层表,用来存储每个用户在各渠道的业务开通情况,主要包括用户号码、开通的业务、渠道以及统计各业务收入的起止日期,具体表的定义如表 10.21 所示。

表 10.21　渠道收入月监控仓库层表

字 段 名 称	字 段 说 明
OP_TIME	数据日期
PRODUCT_NO	用户号码
USER_OPENTIME	用户入网时间
BUSI_ID	业务编码
BUSI_NAME	业务名称
OPEN_DATE	开通日期
CHANNEL_KIND_ID	开通渠道分类
CHANNEL_TYPE_ID	开通渠道类型
BEGIN_CHARGE_DATE	计入渠道收入开始日期
END_CHARGE_DATE	计入渠道收入结束日期

ST_DBIP_CHANNEL_INCOME_MM 是渠道收入月监控应用层表，用来存储各渠道的各个业务的当月收入、历史收入和累计收入，维度主要分为数据日期、品牌和业务，具体表的定义如表 10.22 所示。

表 10.22　渠道收入月监控应用层表

字 段 名 称	字 段 说 明
OP_TIME	数据日期
CHANNEL_TYPE_ID	渠道类型
BUSI_ID	业务 ID
AMOUNT	当月收入
LAST_MONTH_AMOUNT	上月收入
LAST_YEAR_AMOUNT	去年同期收入
ACCUMULATE_AMOUNT	累计收入
LAST_ACCUMULATE	去年同期累计收入

ST_DBIP_PLAN_COMPARE_MM 是对标分析表，用来存储每个月移动集团下发的《增值业务通报表》中的数据，主要包括省份、客户数类指标值、业务的使用量指标值以及用户普及率等，维度主要分为数据日期、省份，具体表的定义如表 10.23 所示。

表 10.23　对标分析表

字 段 名 称	字 段 说 明
OP_TIME	数据日期
BUSI_ID	业务编号
PROVINCE_ID	省份编号
ALL_USERS	移动客户数
ARRIVAL_USERS	到达客户数
ACTIVE_USERS	活跃用户数
AMOUNT	收入
PRICE	单价
ACTIVITY	活跃度
MOBILE_ACTIVITY	手机客户端活跃度
SMS_ARPU	短信 ARPU

续表

字段名称	字段说明
AVERAGE_ARPU	人均 ARPU
AVG_PORTFOLIO	人均使用量
PENET_RAT	用户普及率
USE_PENET_RAT	使用用户普及率
PAY_PENET_RAT	付费用户普及率
CRING_RUNOUT_RAT	彩铃用户流失率
PAY_TO_USE_RAT	付费用户占使用用户比例
MBPHONEPAPER_AMOUNT_RAT	早晚报收入占比
MBPHONEPAPER_USER_RAT	早晚报付费用户占比

ST_DBIP_INCOME_FORECAST_MM 是数据业务收入预测存储表,用来存储每个月实际数据业务收入及每个月的预测数据,主要包括实际收入和预测收入,具体表的定义如表 10.24 所示。

表 10.24 数据业务收入预测存储表

字段名称	字段说明
OP_TIME	数据日期
FORECAST_AMOUNT	预测收入值
REAL_AMOUNT	实际收入值

10.4.3 建模过程

自回归移动平均模型(Autoregressive Integrated Moving Average Model,ARIMA),是由博克思(Box)和詹金斯(Jenkins)于 20 世纪 70 年代初提出的著名时间序列预测方法,所以又称为 Box-Jenkins 模型或博克思-詹金斯算法。其中 ARIMA(p,d,q)称为差分自回归移动平均模型,AR 是自回归,p 为自回归项;MA 为移动平均,q 为移动平均项数,d 为时间序列成为平稳时所做的差分次数。

ARIMA 模型的基本思想是将预测对象随时间推移而形成的数据序列视为一个随机序列,用一定的数学模型来近似描述这个序列。这个模型一旦被识别后就可以从时间序列的过去值和现在值来预测未来值。

AR 模型也称为自回归模型,它的预测方式是通过过去的观测值和现在的干扰值的线性组合预测,自回归模型的数学公式为:

$$\begin{cases} x_t = \phi_0 + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + \varepsilon_t \\ \phi_p \neq 0 \\ E(\varepsilon_t) = 0, \mathrm{Var}(\varepsilon_t) = \sigma_\varepsilon^2, E(\varepsilon_t \varepsilon_s) = 0, s \neq t \\ E(\varepsilon_t \varepsilon_s) = 0, \forall s < t \end{cases} \tag{10.7}$$

其中,p 为自回归模型的阶数,ε_t 为误差,t 为一个时间序列。

MA 模型也称为滑动平均模型,它的预测方式是通过过去的干扰值和现在的干扰值的线性组合预测,滑动平均模型的数学公式为:

$$\begin{cases} x_t = \mu + \varepsilon_t - \theta_1\varepsilon_{t-1} - \theta_2\varepsilon_{t-2} - \cdots - \theta_q\varepsilon_{t-q} \\ \theta q \neq 0 \\ E(\varepsilon_t) = 0, \quad \mathrm{Var}(\varepsilon_t) = \sigma_\varepsilon^2, E(\varepsilon_t\varepsilon_s) = 0, s \neq t \end{cases} \tag{10.8}$$

其中，q 为模型的阶数，μ 为模型的待定系数，ε_t 为误差，t 为一个时间序列。

自回归模型和滑动平均模型的组合便构成了用于描述平稳随机过程的自回归滑动平均模型 ARMA，数学公式为：

$$\begin{cases} x_t = \phi_0 + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + \varepsilon_t - \theta_1\varepsilon_{t-1} - \theta_2\varepsilon_{t-2} - \cdots - \theta_q\varepsilon_{t-q} \\ \phi_p \neq 0, \theta q \neq 0 \\ E(\varepsilon_t) = 0, \mathrm{Var}(\varepsilon_t) = \sigma_\varepsilon^2, E(\varepsilon_t\varepsilon_s) = 0, s \neq t \\ E(\varepsilon_t\varepsilon_s) = 0, \forall s < t \end{cases} \tag{10.9}$$

数据业务收入预测的建模过程如下：

(1) 根据时间序列的散点图、自相关函数和偏自相关函数图以 ADF 单位根检验其方差、趋势及其季节性变化规律，对序列的平稳性进行识别。一般地，经济运行的时间序列都不是平稳序列。

(2) 对非平稳序列进行平稳化处理。如果数据序列是非平稳的，并存在一定的增长或下降趋势，则需要对数据进行差分处理，如果数据存在异方差，则需要对数据进行处理，直到处理后的数据的自相关函数值和偏相关函数值并无显著地异于零。

(3) 根据时间序列模型的识别规则，建立相应的模型。若平稳序列的偏相关函数是截尾的，而自相关函数是拖尾的，可断定序列适合 AR 模型；若平稳序列的偏相关函数是拖尾的，而自相关函数是截尾的，则可断定序列适合 MA 模型；若平稳序列的偏相关函数和自相关函数均是拖尾的，则序列适合 ARMA 模型。

(4) 进行参数估计，检验是否具有统计意义。

(5) 进行假设检验，诊断残差序列是否为白噪声。

(6) 利用已通过检验的模型进行预测分析。

数据业务收入根据历史实际收入数据，应用 ARIMA 预测模型对未来一年的收入进行预测，用户每个月将实际收入作为预测的数据输入，对未来收入进行滚动预测，用户可以对实际收入及预测收入进行查询、对比分析、趋势分析等。收入预测的过程如图 10.8 所示。

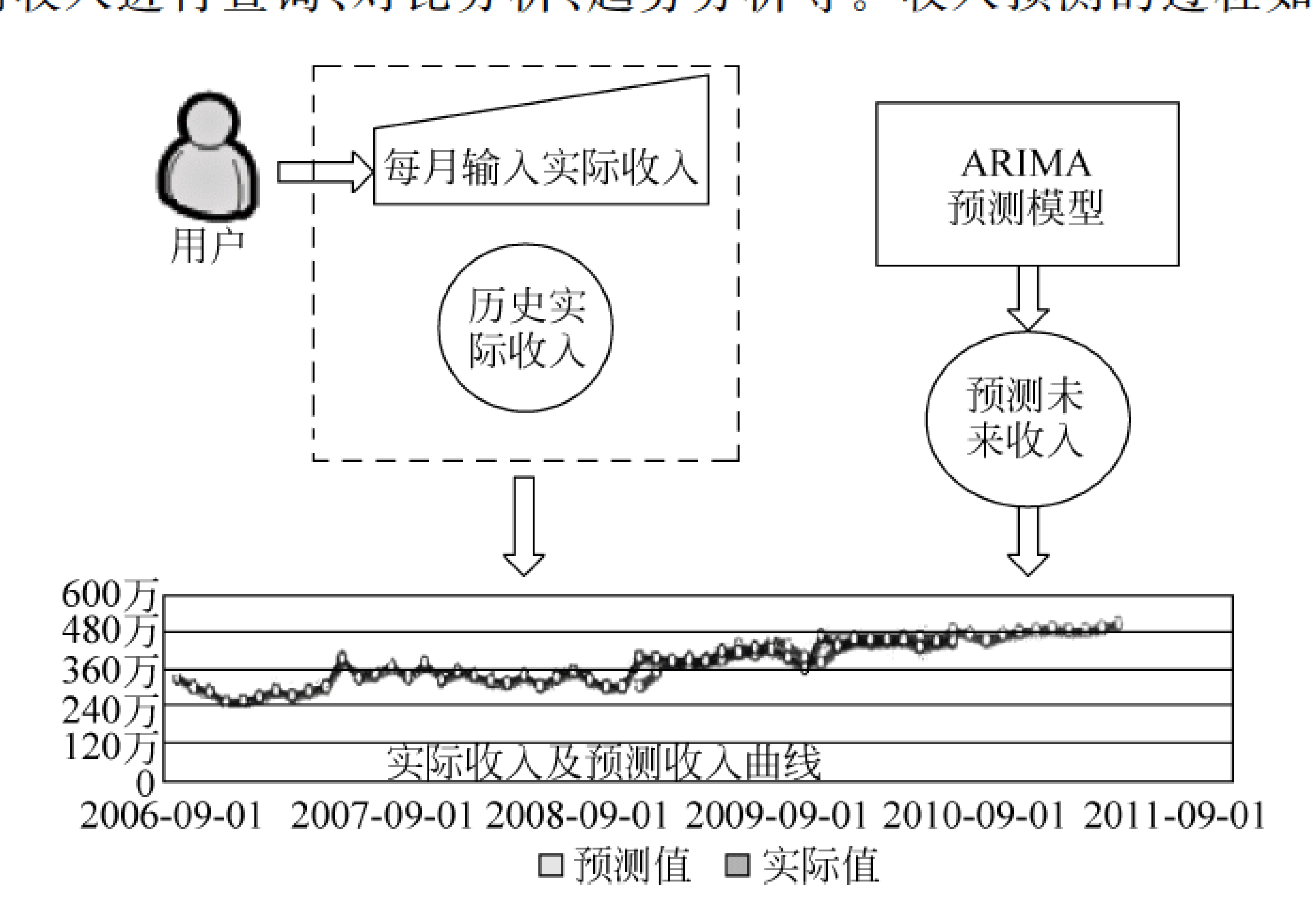


图 10.8　数据业务收入预测过程

10.4.4 结果

利用过去近三年的收入数据和 ARIMA 进行预测，结果如表 10.25 所示，预测的差异率在±10%以内，预测数据波动情况与实际的波动情况基本一致(如图 10.9 所示)，因此采用此预测方法基本上可以满足用户的要求。

表 10.25　收入预测值与实际收入对比表

数据月份	预测值(元)	实际值(元)	差异(元)	差异率
2009-01	89 504 684.36	8 7679 674.99	1 825 009.37	2.08%
2009-02	102 022 149.60	100 516 365.52	1 505 784.08	1.50%
2009-03	110 190 192.56	112 074 780.47	−1 884 587.91	−1.68%
2009-04	108 368 918.28	113 752 227.15	−5 383 308.87	−4.73%
2009-05	113 922 249.28	112 066 256.88	1 855 992.40	1.66%
2009-06	113 694 142.24	121 229 181.07	−7 535 038.83	−6.22%
2009-07	127 078 139.16	120 197 891.71	6 880 247.45	5.72%
2009-08	118 309 652.16	123 953 333.43	−5 643 681.27	−4.55%
2009-09	133 326 950.96	123 003 491.86	10 323 459.10	8.39%
2009-10	125 804 930.16	115 409 303.87	10 395 626.29	9.01%
2009-11	115 750 794.88	105 475 040.64	10 275 754.24	9.74%
2009-12	110 393 109.96	118 343 182.47	−7 950 072.51	−6.72%
2010-01	127 225 535.92	126 082 499.80	1 143 036.12	0.91%
2010-02	130 903 640.00	133 029 303.64	−2 125 663.64	−1.60%
2010-03	129 521 499.24	132 113 160.21	−2 591 660.97	−1.96%
2010-04	130 807 013.68	131 683 668.34	−876 654.66	−0.67%
2010-05	134 786 825.88	131 559 150.39	3 227 675.49	2.45%
2010-06	134 297 487.24	125 109 544.96	9 187 942.28	7.34%
2010-07	130 945 950.52	129 880 845.12	1 065 105.40	0.82%
2010-08	140 447 183.24	130 581 895.36	9 865 287.88	7.55%
2010-09	136 699 384.64	139 433 372.33	−2 733 987.69	−1.96%
2010-10	130 589 300.80	127 977 514.78	2 611 786.02	2.04%
2010-11	136 675 625.80	129 841 844.51	6 833 781.29	5.26%
2010-12	139 362 168.68	149 117 520.49	−9 755 351.81	−6.54%
2011-01	141 138 069.52			
2011-02	142 029 118.84			
2011-03	140 375 757.20			
2011-04	140 684 772.76			
2011-05	142 627 225.16			
2011-06	146 261 889.76			
2011-07	145 105 681.00			
2011-08	154 970 731.57			
2011-09	151 904 763.86			
2011-10	170 300 570.13			
2011-11	168 920 884.66			
2011-12	170 021 845.79			

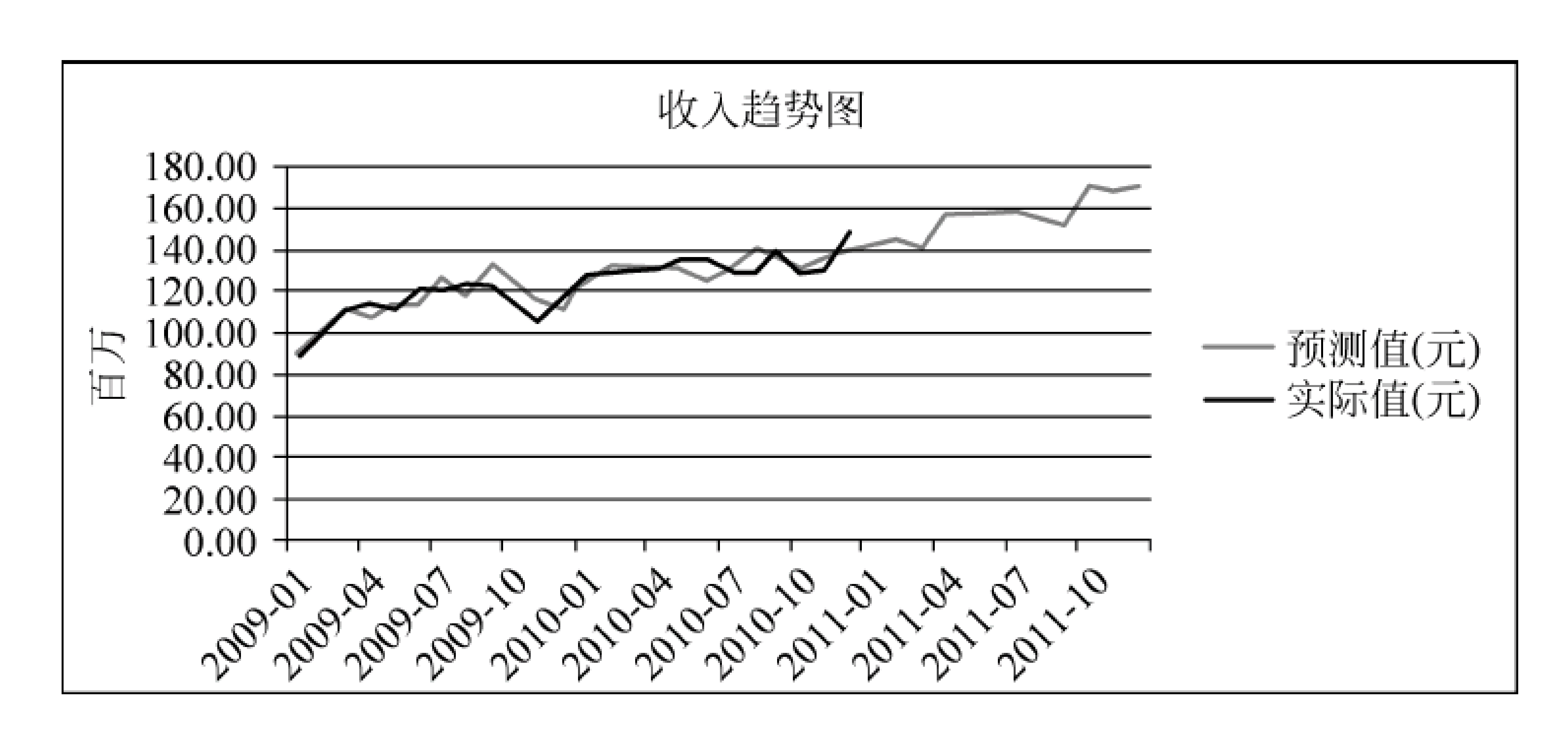


图 10.9　数据业务收入预测结果

10.5　移动客户流失预测

客户流失已经成为很多企业所面临的尴尬,他们大多也都知道失去一个老客户会带来巨大的损失,需要企业至少再开发 10 个新客户才能予以弥补。但当问及企业客户为什么流失时,企业往往一脸迷茫,谈到如何防范,更是诚惶诚恐。如今,市场竞争日益激烈,谁保持住了老客户,谁就能在竞争中占据上风。

移动通信的客户流失是一个长久以来困扰全球移动通信运营商的难题。在欧洲,每年有 35%～50%的客户流失,而发展一个新客户的平均成本为 70 元。这几乎相当于一个客户 5 年来给公司带来的净利润。这种情况直接导致客户回报率的下降。在美国,不少移动通信运营商有着 40%的流失率,而发展一个新客户的成本超过了 40 元。在亚洲的日本、中国的台湾和香港等地,移动通信市场正在经历过激竞争阶段,伴随而来的是极高的客户流失率。从全球范围来看,移动运营商每年因为客户流失所带来的损失在 100 亿元左右。

早期的移动通信市场营销战略重点在市场开拓和客户发展上,对于客户流失没有引起足够的重视,对可能流失客户的维系与挽留措施不到位。随着运营商的技术和产品同质化程度的不断提高,入网费用逐步下调甚至取消,转网成本迅速降低。出于竞争的目的,各运营商又纷纷出台促销套餐,使得客户更换号码,在网间转来转去的情况较为普遍,形成所谓的"旋转门效应"(Revolving-door Effect),即新客户进来时原有的客户却出去了,像旋转门一样的进进出出。这种"大进大出"现象使运营商经营成本大幅上升,为了发展一个新客户,需要付出高昂的成本。营销实践表明新增客户的 ARPU 值(月平均消费额)一般不及老客户。因此,对于电信运营商而言,分析客户流失原因,提前预测客户离网倾向,并有的放矢地采取有效措施降低离网率,具有重要的现实意义。具体地,可归纳为以下三个方面。

第一,可以避免与竞争对手打价格战。客户流失率直接影响到运营商的市场占有率、竞争成本和经济效益。而一旦客户离网,一般会简单地采取直接降低资费、推出更多的优惠进行市场回应,并试图一次性地快速控制过高的流失率,而资费的降低是不可逆的,最终将导致竞争双方被迫陷入价格战的旋涡,形成价格越来越低,收入越来越少的恶性循环。

第二,可以降低营销成本。通信企业发展新客户将投入大量的广告费、促销费、代理费及 SIM 卡等一系列费用。因此,挽留一个老客户的意义不亚于发展 3～5 个新客户的意义。

第三,可以与客户建立稳定联系,延长客户生命周期,提升其忠诚度及价值,进行增量或

交叉销售,进一步推广新业务和增值业务。

目前,新入网的移动客户中以低端为主,高端客户的增长主要来源于竞争对手或运营商内部不同品牌的转换,高端客户市场基本处于存量竞争状态。与固定电话客户相比,移动业务客户转换运营商的交易成本和转换成本较小,激烈的竞争必然导致客户处于不稳定状态,各运营商都面临严重的客户流失问题。客户的平均生命周期不断缩短,严重影响电信企业的发展,不仅给运营商的收入造成损失,也给企业的经营带来压力。在这种形势下,如何根据客户各种行为预测客户的流失,并结合竞争环境的变化制定有效的客户挽留和赢回策略,力求未雨绸缪,是一个迫切需要解决的问题。

10.5.1 定义

客户流失是指客户因某种原因与运营商解除服务合同的行为。移动业务的客户流失有三方面的含义:一是指客户从一个运营商转到其他运营商,这是流失分析的重点;二是指客户使用的手机品牌发生改变,从本运营商的高价值品牌转向低价值品牌,例如中国移动的客户从全球通转为神州行;三是指客户 ARPU 降低,从高价值客户转为低价值客户。

此外,可以根据流失原因,如价格、服务质量、公司形象和竞争对手等因素将客户流失划分为自然流失、恶意流失、竞争流失和过失流失四种类型。

1. 自然流失

自然流失不是人为因素造成的,如客户的搬迁、死亡等,其流失数量不大,而且对企业的影响也较小,不需要重点分析。

2. 恶意流失

恶意流失是指一些客户为了满足自己的某些私利而选择离开,最典型的例子就是恶意欠费。这类客户的行为应该属于一种欺诈行为,相对较少,但是性质恶劣,需要严加控制。对于这类客户造成的问题主要还要从预防恶意高额欠费和信用监控等方面着手解决,建立完善的客户信用管理机制,避免该类流失客户的增多。

3. 竞争流失

竞争流失是由于企业竞争对手的影响造成的,竞争焦点主要集中在价格、质量和服务方面。由于竞争对手在各方面对客户都更具有吸引力,从而导致客户流失到竞争对手。企业可以采取相应策略,提高自身的服务质量挽留客户。

4. 过失流失

过失流失大都是由于企业自身过失造成的,或者是一些决策失误导致的客户离网,这部分流失客户占流失总量的比例最高,带给企业的影响最大。例如欠费停机所采取的策略问题,很多客户只是由于暂时没有来得及缴费导致欠费停机,而有的客户是恶意欺诈导致高额欠费停机,应该说是两种欠费停机的性质完全不同,而相关策略对于两者并没有区别对待,导致某些客户流失。如果区别对待,则可能挽留住有价值的客户,提高企业收益。

目前,客户流失预测的常用方法之一是利用大量的历史数据通过建立流失预测模型找出隐含的业务规则,并运用已经学习训练好的模型预测未来可能流失的客户。

10.5.2 数据准备

数据准备阶段的工作主要是选择合适的数据源，整理分析所需的信息，包括通话、消费、欠费和客服行为数据等。然后，对原始数据进行检查和预处理，生成新的衍生变量，并整理成适合分析使用的数据宽表。因为并不是所有的信息都会对客户流失产生影响，应尽可能地降低数据的复杂度以挖掘较高的关联度，但是考虑到后期客户流失的多维分析，应当尽量确保客户信息的完整性。因此，应收集有价值的客户信息，剔除部分冗余数据，减少噪声数据。

具体的数据准备过程如下：

1. 选择数据分析范围

根据业务需求选取某本地网 35 731 小灵通客户最近连续四个月的数据，以前三个月数据作为训练集，第四个月数据作为测试集，即利用最近连续三个月的数据，预测未来一个月客户是否流失。

2. 收集原始数据

从计费账务系统、97 系统、10000 号管理中心、关口局等抽取大量详单级数据。

3. 数据清洗

将获取的数据经过 ETL 过程导入到本地数据仓库的过程可能会遇到一些问题，如 10000 号的数据是日志格式的，需要先用 SQL 编程实现从日志到详单的转换，然后再计算并汇总详单；很多客户资料与计费账务系统无法一一对应；关口局的详单数据需要 ETL 的进一步归并等。非常繁琐而且运算量很大。因此，需要事先明确数据清洗的规则。

4. 检查数据质量

在检查前面提取的数据时发现数据存在如下问题：

- 数据缺失严重，绝大多数的客户属性都有大量的缺失值，例如账务系统中的 200 多个属性，很多客户都是 null，业务标识也是如此，由于总体缺失值比例过高，根本没办法进行合理的缺失值处理，所以根据客户流失的业务定义剔除一些无关的属性。
- 合同号所属电话号码月租分布差异很大，将月租属性单独列项。
- 各地合同号编写规则不一，需统一规则。
- 并网导致同一合同号所属区域矛盾，需归账统一。
- 出账费用有些小于零的情况，属于已经享受优惠套餐或冲减话费应剔除。

5. 计算衍生变量

根据客户流失预测的建模需求，需要计算一些衍生变量，包括总费用趋势、总费用波动、本地费用波动、本地费用趋势、长途费用波动、长途费用趋势以及网间、欠费和客服行为等。

6. 合并生成宽表

按照客户标识号合并各种数据，生成一张数据宽表，其中每行代表一个客户，每列代表一个变量。

10.5.3 特征变量选取

在上述的数据准备中，针对客户行为数据，进行了数据的抽取、整理和合并等，生成了分

析所需的数据宽表。所形成的数据宽表中,包含226个变量,除了其中10个是客户属性信息外,其余216个变量都是行为特征数据,这些变量可能对流失有影响,也可能没有。如果把所有变量都纳入分析会降低效率,而且无关变量还可能引起结果的误差。因此首先应该对数据宽表中的各变量进行相关性分析,找出每个变量与客户流失概率的相关性,删除相关性较小的变量,尽可能地降低数据和模型的复杂度。

特征变量筛选的主要方法包括如下几种。

1. 数据质量校验

经过对原始数据的质量校验发现部分客户的资料表与消费行为表的属性对应不上,造成少量客户的消费行为数据无法获取。对数据宽表中的字段进行质量分析,发现部分变量的有效性较低,数据无效是由于在生成分析变量时相除运算的分母为零,这种情况下的有效性低表明绝大多数客户在该变量上取值相同,此时该变量对建模无实际意义,不纳入分析。通过对数据质量的校验发现其他费用趋势、其他费用波动、网间长途与IP时长比例、网间长途与IP次数比例等变量的数据有效性低于20%,在建模时不再作为分析变量。

2. 流失结果初步分析

从整理出的数据宽表中,可以对训练集的流失结果进行初步观察。

3. 考察数据的分布情况

考查数据宽表中所列变量的分布,从数据分布状况再次查看数据质量。从中可以发现异常值的状况、数据的离散趋势等。对于异常值需要分析其产生原因,并判断是否需要修正或剔除;对于数据离散度很低的变量,可以考虑不纳入分析变量。

如在总费用中发现最小值为负,表明存在异常,该异常是由于退费等原因引起的,这种情况属于极少发生的个别行为,在建模时可以考虑剔除。又如观察数据离散趋势时,可以重点考察其标准差、平均值等统计量。标准差反映了变量取值的整体偏差,如果偏差很小,则表明数据的取值非常接近。因从数据中发现港澳台全天通话次数、港澳台全天时长、区间全天时长、区间通话次数、市话次数比例、本地费-区间、港澳长途金额和故障次数等数据偏差非常小,故不纳入分析。

4. 考查变量对流失的影响

要研究客户的流失可能性,就需要找出对流失结果有重要影响的变量,降低数据计算的复杂性。把流失和不流失作为两个分组,对生成的变量在这两个组的差异进行比较,差异大的应作为建模变量,差异小的对建模意义不大。根据差异的显著性程度,进行变量选取。

研究变量的组间差异的方法很多,如直方图的方法。由图10.10可知,大部分用户的通话时长较小,时长越大的客户,流失概率越低,直观可知该变量对流失是有影响的。

5. 考查变量间的关系

数据宽表中包含的变量很多,在建模之前需要了解它们之间的相关性。对于关系十分密切的变量,在建模时可以选择其中最具代表性的变量,这样不但会大大提高运算效率,更可减低变量冗余造成结果偏差的风险。根据经验,通话行为数据中工作时段、非工作时段、周末时段的综合应该为全天的统计值,故在分析时可以去掉其中之一。

经过数据准备和特征变量筛选后,数据可基本满足客户流失预测建模的要求。

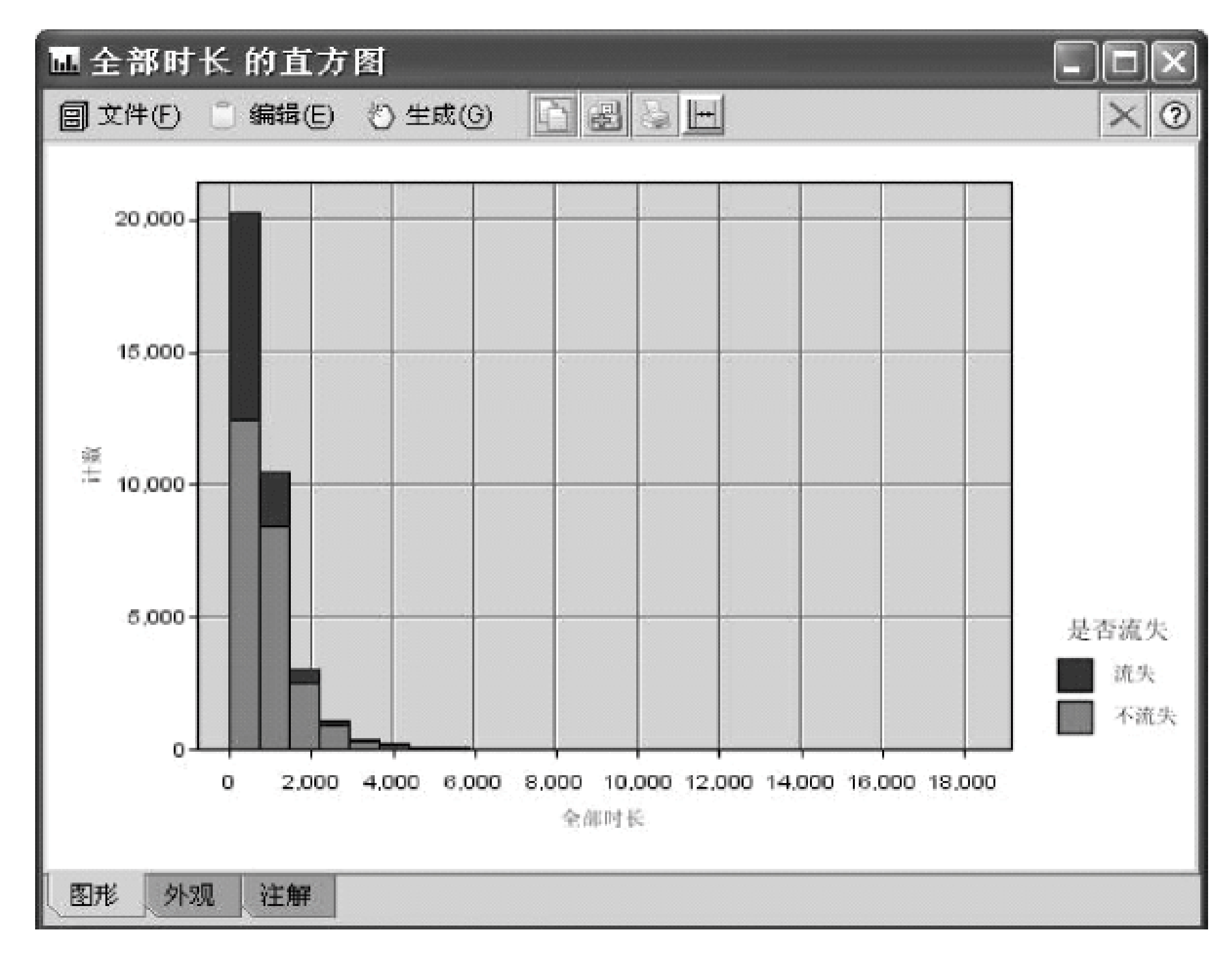


图 10.10　全部通话时长的直方图

10.5.4　建模过程

下面介绍客户流失预测的建模过程。

1. 模型的选择

现有的数据挖掘工具提供了决策树、神经网络、回归、关联、聚类和贝叶斯等多种方法。可以分别使用多种方法建模，然后对这些模型进行比较，从而挑选出最适合客户流失预测的方法。此外，数据挖掘工具还提供了选择建模方法的功能，可自动判别最优模型，供使用者参考。

通过特征变量筛选以及实际经验，最终确定利用总费用、区内费用、区间费用、国内长途、国内 IP、增值业务与信息费及以上变量的波动和趋势变量等进行流失预测建模，如表 10.26 所示。

表 10.26　客户流失预测的特征变量

名　称	描　述	逻　辑
主 ID	电话号码	设备终端号
总费用	当月总费用	某月份客户总费用(优惠后费用)
区内费用	当月区内费用	某月份客户区内费用
区间费用	当月区间费用	某月份客户区间费用
国内长途	当月国内长途费用	某月份客户国内长途费用
国内 IP	当月国内 IP 费用	某月份客户国内 IP 费用
总费用趋势	过去 3 个月平均值减当月值除以过去 3 个月平均值加上当月值的商	$\dfrac{\frac{X_1+X_2+X_3}{3}-X_4}{\frac{X_1+X_2+X_3}{3}+X_4}$
区内费用趋势		
区间费用趋势		
国内长途趋势		
国内 IP 趋势		
增值业务趋势		
信息费趋势		

续表

名　称	描　述	逻　辑
总费用波动	过去3个月和当月的4个值中的最大值减最小值除以平均值	$\frac{4[\mathrm{Max}(X_1,X_2,X_3,X_4)-\mathrm{Min}(X_1,X_2,X_3,X_4)]}{X_1+X_2+X_3+X_4}$
区内费用波动		
区间费用波动		
国内长途波动		
国内IP波动		
增值业务波动		
信息费波动		
在网时长	客户在网月份数	某月份账期结束时客户在网月份数
当月缴费额	当月缴费总额	某月份客户缴费总额
联系人数	当月呼叫不同号码数	某月份客户呼叫不同号码个数
标识	流失客户标识	流失客户,标1,否则标0

分别采用C5.0和神经网络两种算法,由表10.27可知,就正确率而言,神经网络模型预测客户是否流失的正确率较高,所以作为客户流失预测的模型。

表10.27　C5.0和神经网络两种模型的客户流失预测正确率

	测试样本数量	预测正确样本数	预测错误样本数	正确率
神经网络	10 000	8134	1866	81.34%
C5.0	10 000	7383	2617	73.83%

2. 模型建立与调整

模型建立与调整是数据挖掘过程的核心部分,通常由数据分析专家完成。需要指出的是,不同的商业问题和不同的数据分布属性会影响模型建立与调整的策略,而且在建模过程中还会使用多种近似算法来简化模型的优化过程。因此还需要业务专家参与调整策略的制定,以避免不适当的优化造成业务信息丢失。

3. 模型的评估与检验

应该利用未参与建模的数据进行模型的评估,才能得到准确的结果。检验的方法是使用模型对已知客户状态的数据进行预测,将预测值与实际客户状态进行比较,预测正确率最高的模型是最优模型。

4. 模型解释与应用

业务人员应该针对最优模型进行合理的解释。如发现开户时长与客户流失概率的相关度较高,利用业务知识可以解释为客户在使用一定年限后需要换领新SIM卡,而这一手续的烦琐导致客户宁愿申请新号码,从而造成客户流失。通过对模型做出合理的业务解释,可以找出一些潜在的规律,用于指导业务行为。反过来,通过业务解释也能证明数学模型的合理性和有效性。

在模型应用过程中,可以先选择一个试点实施应用,试点期间随时注意模型应用的收益情况。一旦发生异常偏差,则立即停止应用,并对模型进行修正。试点结束后,若模型被证明应用良好,可以考虑大范围推广。推广时应注意,由于地区差异,模型不能完全照搬。可以先由集团总部建立一个通用模型,各省分公司在此基础上利用本地数据进行修正,从而得

到适用于本省的精确模型。在模型应用一段时期，或经济环境发生重大变化后，模型的偏差可能会增大，这时应该考虑重新建立一个适用性更强的模型。

10.5.5 结果

首先，利用 35 731 个训练样本对客户流失预测模型进行训练，训练结束后形成了 10 个分类的流失概率，并由业务人员根据各类别特征描述其业务特征，如表 10.28 所示。

表 10.28 流失预测的结果

类别	客户数	类别描述
类别 1	7821	市话多，短信少。
类别 2	335	使用频率很低，总费用少，一般只在周末打打电话。
类别 3	2700	优质客户，朋友多，本地和长途通话都比较多，尤其网间长途多。
类别 4	5322	优质客户，本地长途费都高，短信费也较高。
类别 5	4074	一般客户，总费用偏低。
类别 6	99	高级客户，平均通话时间特别长，网间长途特别多，短信费高。
类别 7	113	主要交月租，只用来接听电话。
类别 8	10 227	一般客户，总费用较高。
类别 9	4342	短信较多。
类别 10	698	朋友不多，通话量较少。

然后，利用测试集进行测试，对每个客户的流失概率给出一个评分，即该客户下个月的流失概率，部分结果如表 10.29 所示。

表 10.29 流失预测的部分结果

客户 ID	用户 ID	号码	预测流失率
51222893	3892381	8×××000	0.86
51149757	3757337	8×××005	0.90
51276654	4447622	8×××008	0.50
51149756	3757335	8×××010	0.90
51073165	3764659	8×××055	0.57
51123422	3785384	8×××056	0.90
51127025	3917040	8×××057	0.58
51149814	3764597	8×××058	0.91
51122864	3764598	8×××059	0.90
51123477	3786508	8×××060	0.90
51151746	3882974	8×××062	0.86
51151721	3882457	8×××063	0.66
51126583	3907612	8×××064	0.50
51151720	3882455	8×××065	0.50
51122948	3764747	8×××066	0.50
51074420	3906954	8×××067	0.91
51246552	3885516	8×××070	0.92

根据流失预测的结果计算出不同客户占比的模型评估的覆盖率、准确率以及提升率，如图 10.11～图 10.13 所示。

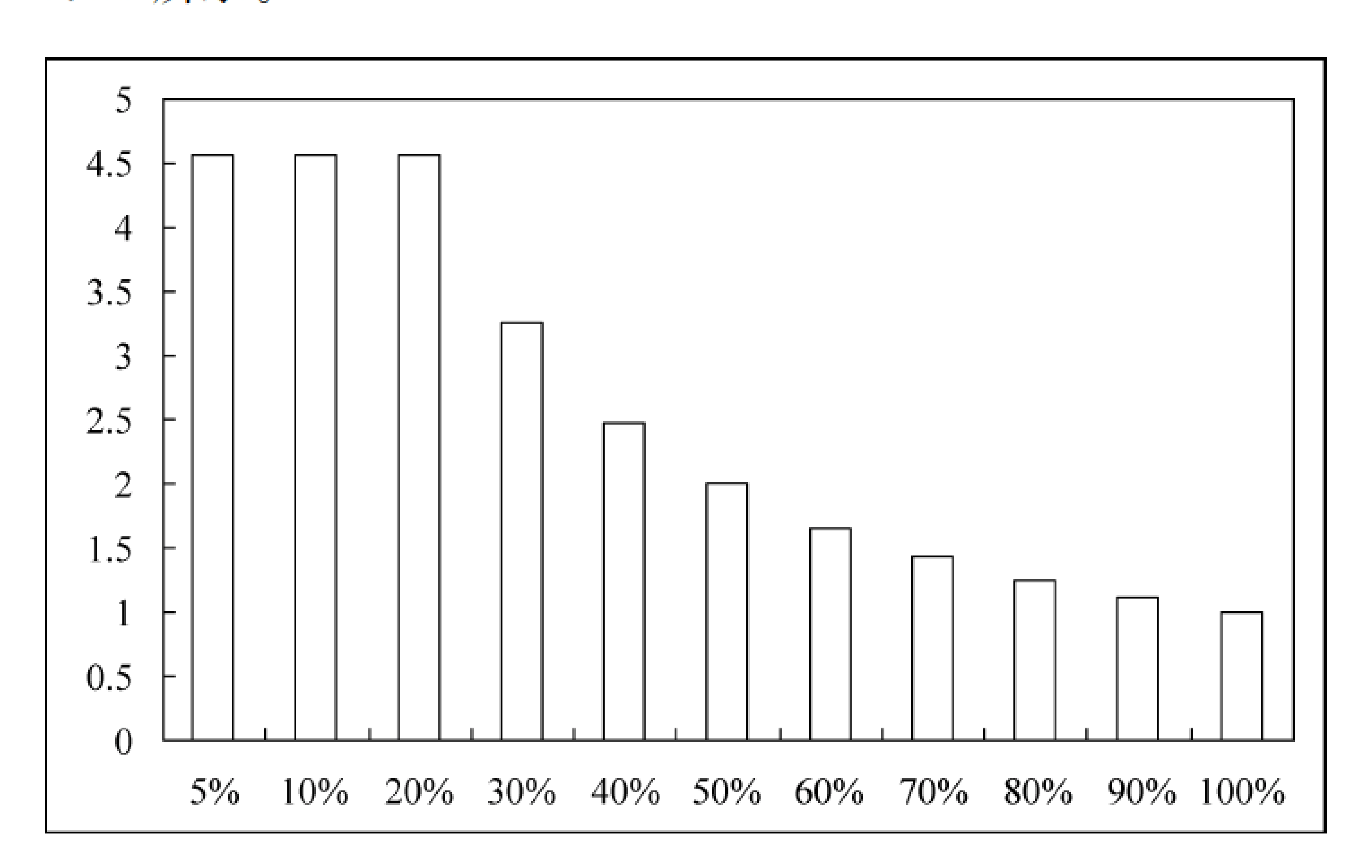


图 10.11　提升率分布示意图

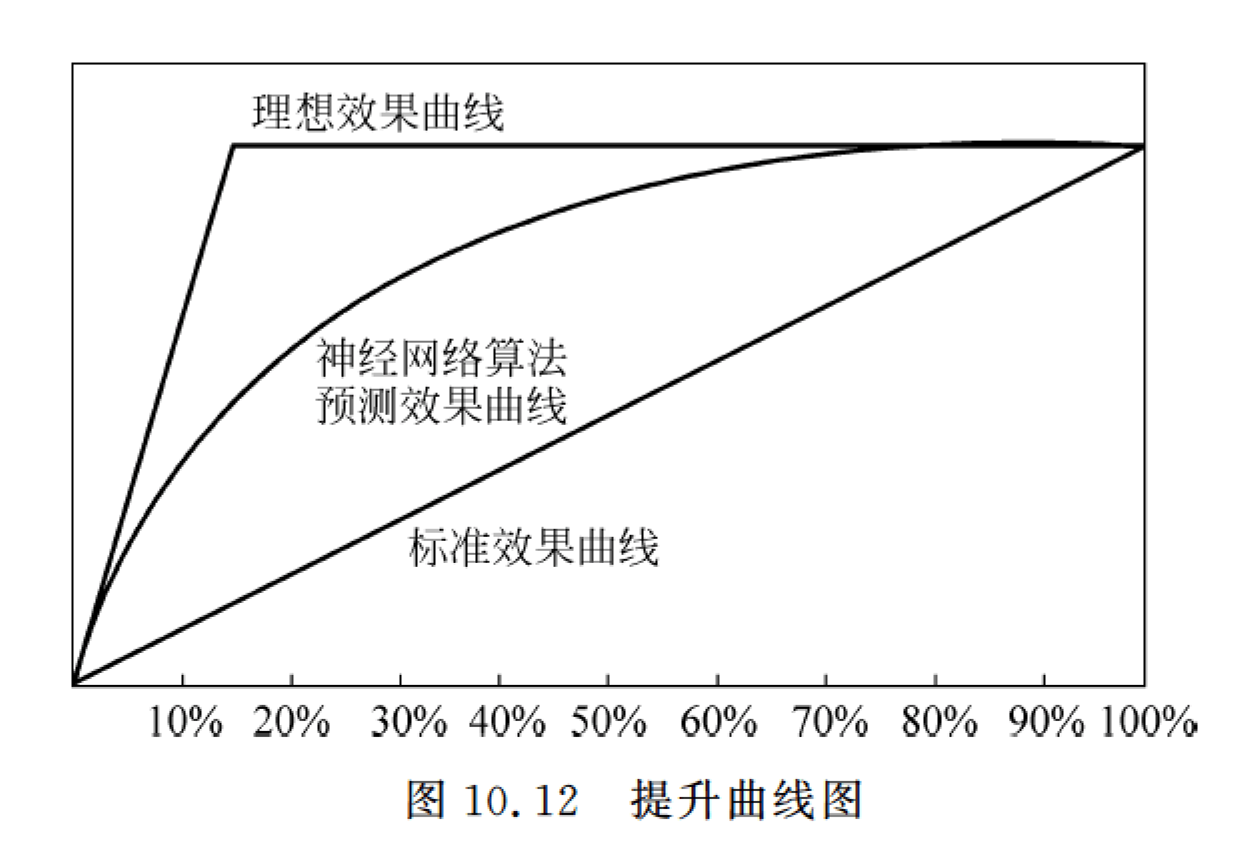


图 10.12　提升曲线图

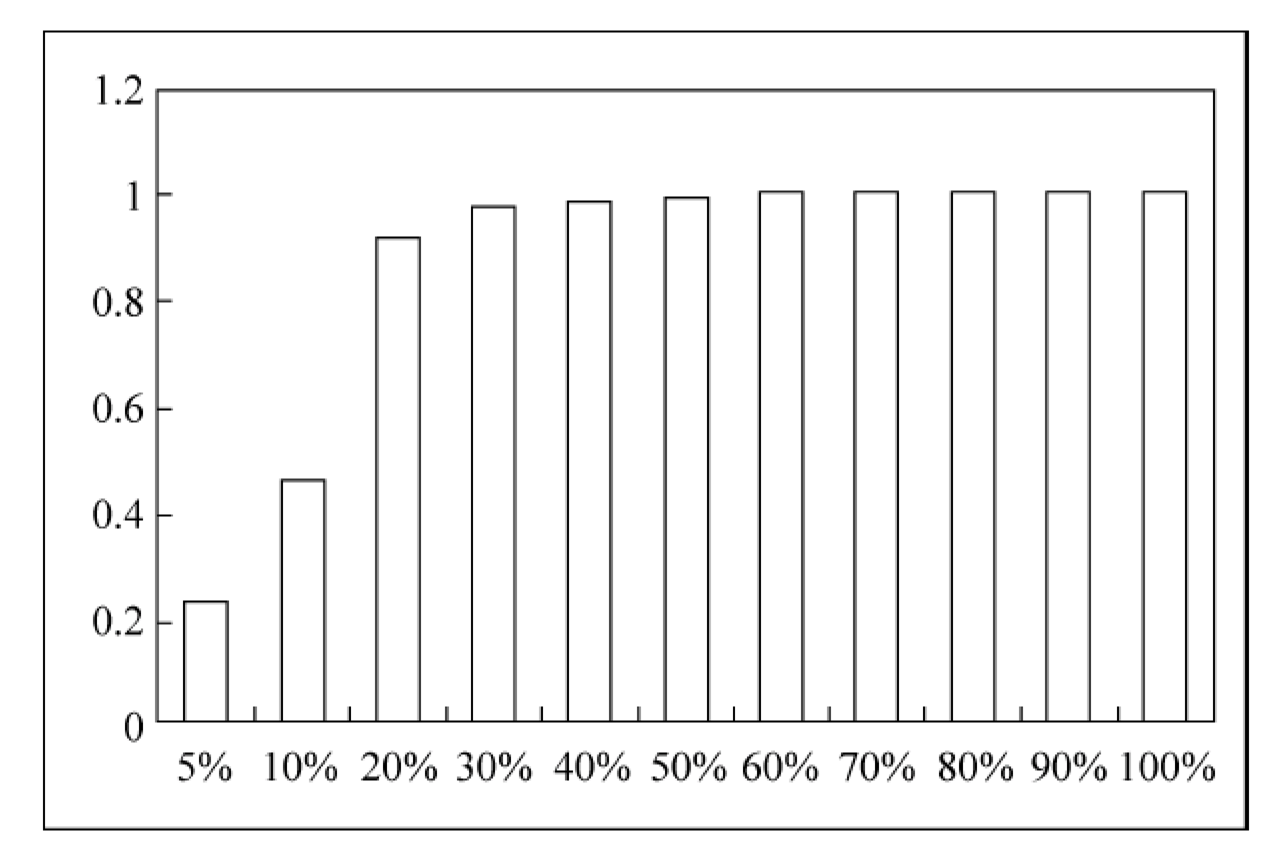


图 10.13　覆盖率分布示意图

从各项指标来看，模型的效果较好。例如，前 5%的客户的流失覆盖率为 22.82%，准确率为 99.97%，说明通过上述过程建立的客户流失预测模型预测每 100 个客户可以正确识别 99.97 个，而正常的浓度是 100 个客户中有 22 个，而前 20%的客户中就将有 90%以上的

是即将流失的客户。根据预测结果按照对评分高低给出的客户名单进行营销，可以节省营销成本到原来的五分之一，在降低营销成本的同时提高了客户营销的成功率，达到了客户流失预测的目的。

在此基础上，营销管理人员调用预警汇总表进行营销派单，根据客户实际流失情况对模型进行评估，各级领导查询客户流失预警情况的报表，客户经理调用所负责客户的预测情况以及将实际结果维护到预警表中，营业受理人员查询前来办理业务的客户流失预测情况，以便利用业务受理机会实现及时的面对面的主动营销等。

尽管客户流失的原因多种多样，流失原因差异较大。综合上述的客户流失因素的定量分析以及抽样问卷调查结果，可将客户流失原因归纳为以下六个方面。

1. 经营管理

1）客户关系管理人员配置不到位，客户关系管理工作缺乏深度和内涵。直接影响到客户感知，增加了客户流失的概率。

2）政策不完善让代理商有机可乘，如不收取 SIM 卡费，支付给代理商的首次酬金过高，让代理商可补贴的优惠范围大，使新入网用户入网无任何成本、入网门槛过低，导致在网低价值客户增加。针对新发展客户制定出一些专门的优惠措施，而在网老客户却不能享受，造成新、老客户的优惠不平衡，导致老客户弃卡离网。

3）在以放号量为导向的绩效考核下，部分县分公司在一定时期内还是会依赖一些有跨区放号能力的批发商，在分配号源时也会倾向于这些代理商，批销代理商在逐级批发的过程中提前用虚假客户资料进行开通，导致了系统内大量虚开客户。虚假的客户数据对于客户欠费缺乏足够的证据进行追讨，为客户欠费弃卡提供了便利。

4）为了不断提升客户的优惠感知，不断推出的促销活动，加之市场区隔的不明显，让前后参与活动的客户满意度降低，导致离网。

2. 渠道管理

1）服务缺失

由于长期以来渠道管理被相对忽视，故在人员配备和职能建设上相对滞后。渠道管理人员的缺乏将直接影响到渠道的监管效率，渠道管理的职能仅限于提供代理商领卡、酬金审核一些极有限的工作。对渠道主动管理，如宣传资料配送、业务知识培训、巡检、管理考核基本上是一种缺失状态。社会渠道基本业务知识了解不够全面，业务告知、业务推荐不准确，营销活动很难迅速渗透到社会渠道。

2）管理粗放

对于社会渠道的管理，长期以来采用的是根据渠道类型分层管理的方式，然而在具体管理形式上则缺乏“精细化管理”，其具体表现在对社会渠道仍然使用相对单一的激励手段。但对其服务水平提升，管理规范等方面则缺乏效果，并且容易造成经销商的短期逐利行为，但无法对其形成捆绑效果，其忠诚度的巩固也缺乏有效的激励体系保证。

3）经营投机

通常运营商放号酬金的支付分为三次，第一次代理商在领号即可返给酬金比率达 70％的领号酬金，首次酬金过高；第二次为在网三个月后；第三次为在网半年后。首次领号酬金的支付方式导致各层级批发商在逐级批发中都有可观的利润空间，而最顶端的代理商只需要拿到二次或三次在网酬金就已足够丰厚了。这样也就造成在逐级批发前批发商需将号

码预先进行虚开,而在逐级批发的过程中各层级代理商将酬金补贴至号码中低面值销售,诱导客户弃卡重新入网。

3. 竞争

竞争体现在策反、促销优惠和品牌服务等方面。

1) 策反

在通信市场的激烈竞争格局下,各运营商之间的竞争也趋于多样化,主要包括:

(1) 宣传竞争。利用客户生活质量提升,追求自然、时尚、健康的消费理念,例如CDMA大力宣传环保、绿色、常打头不痛等。

(2) 客户质量的竞争。各运营商均投入大量的资源开展针对高端客户、法人团体客户有目的的拉拢、回挖。

(3) 渠道的竞争。得渠道者得天下,谁先掌握了渠道,谁就掌握了市场竞争的主动权。针对一些大的代理商或非排他性代理商利用高放号酬金及客户在网一年之内的话费分成进行拉拢,而代理商受利益驱使,造成渠道阶段性的客户推荐导向,以从中牟利,并对这些代理商掌握的客户资源进行回挖。

2) 促销优惠

在激烈的市场竞争下,各运营商均在大力投入成本开展各式各样的市场营销,营销活动中新老客户优惠不平衡,为了促进新增,花了很大的营销成本,对三个月以上的在网客户基本上没有任何营销优惠,销号、新入网成本为零,甚至小于零,导致客户频繁换号。

3) 品牌服务

各品牌之间的服务区隔不明显,客户对品牌的价值感知不强。且频繁的价格变动及促销优惠,在吸引新入网客户的同时,必然损害老客户的利益,从而导致老客户的品牌忠诚度下降,资费便宜仍是客户购卡的主要驱动因素之一,客户品牌感知度较低,客户品牌意识不强,对移动产品的理解仍停留在资费、套餐上。最终导致客户改为低资费套餐或受竞争对手低资费吸引而转网。

4. 产品质量

产品质量体现在网络覆盖率、通话质量、业务提供和终端质量等多方面。

1) 网络覆盖

虽然总体上而言,网络覆盖较好,但在一些偏远的农村还存在盲点。

2) 通话质量

由于载频不足、新建站容量不足、新建站入网慢、话务集中、城区楼宇密集导致覆盖盲区、客户活动区域大导致通话质量波动还有客户手机终端质量问题等导致客户对通话质量不满,导致客户满意度降低而离网。

3) 业务提供

由于系统支撑问题,客户使用的资费套餐不能实现网内互转,而促使客户重入网选择新的资费套餐。

4) 终端质量

由于手机终端质量、性能不一,客户对终端质量产生的通话质量感知不一,加之竞争对手的低端购机及价格战的吸引,导致客户转网。

5. 服务水平

20世纪80年代,影响消费理念的决定性因素是价格;到了20世纪90年代,人们的消

费理念由价格转向追求具有卓越品质的优质产品；随着社会的不断发展进步，人们的消费理念也发生了根本的改变。今天，人们的消费理念除价格、品质外，又增加一项全新的需求——服务。服务是品牌维护社会形象，赢得市场信誉，建立客户忠诚度的基础。服务质量的好坏，将直接影响品牌的价值，据营销机构的市场调查分析，服务好的产品比服务差的产品售价高出10%，而营业额、客户规模、客户忠诚度却增长较快，市场占有率年增6%，销售利润增12%。从某种角度来看，消费者在选择同类产品消费时，已把商家是否建立了较优良、完善的一体化服务体系作为较重要的衡量标准。

客户面对三家运营商，可选的服务较多，因此出现更加频繁的离网行为。一般而言，离网客户可分为外部离网（销号）、内部转网和沉默（基本没有消费产生）三种。由于移动通信市场资费差异渐渐缩小，服务对客户的影响越来越大，不愉快的服务经历、服务功能的缺失都有可能导致客户离网，而且由于服务原因离网的客户更难挽留，一旦离网要让其回网更加困难。

服务水平包括计费准确性、服务的手段、态度和效率等多方面。

1）计费误解/误差

客户资费、账单收取错误或客户对资费套餐理解错误，但营销服务前台并未给客户解释清晰，使客户对计费产生不信任感，从而导致客户离网。客户对预存业务按月返款理解不清晰，导致客户未及时缴费，欠费销号后系统将不继续返款，从而导致客户离网销号。随着预存营销业务的增多，此部分客户呈现增长趋势。

2）服务人员态度/效率

服务人员仅有服务意识，并未从心理上、行为上去自发为每个客户提供优质服务。个人综合素质差异，公司业务流程设计的复杂性及系统的有力支撑均影响到服务人员的效率。

3）终端售后

在营销活动开展后，对客户的使用跟进不到位，未能及时处理客户的终端售后，特别是低端机营销，客户便会弃卡重新入网。

6. 客户行为

客户离网的原因复杂多样，各种离网原因之间存在交叉，有的甚至互为因果，按离网行为方式可分为主动离网和被动离网。

1）主动离网

主动离网的原因主要包括：

对费用敏感。一是增值业务的定制取消流程复杂不透明，客户因高额增值业务费用被迫销号；二是产品选择不当，对产品互转不了解或产品互转受限制，便选择重新入网；三是客户希望使用的资费套餐不能带号转入，促使客户重入网加入新的资费套餐；四是长途、漫游资费较高而产生离网；五是新入网优惠或竞争对手更加优惠，新入网成本远远低于继续使用现有产品，引发客户离网。

业务服务提供不到位。一是业务办理流程复杂，业务办理门限导致客户弃卡新入网；二是客户满意度降低如对投诉处理不满意或是服务提供便利性、及时性不是很到位，导致客户离网；四是在农村偏远区域存在缴费难、欠费易，弃卡入网更实惠，客户为了逃避欠费而离网的现象是较为明显。

此外，还有一些其他原因。一是流动人员离开本地，客户换号产生离网；二是对客户因手机丢失/损坏，对业务不了解、不熟悉而只知道新购卡而重新入网或者是受新入网优惠的

吸引产生离网;三是号码不理想,换号产生离网;四是社会代理渠道鼓励换卡离网。

2）被动离网

被动离网的原因主要是欠费销号,根据主观意愿也可分为主动和被动,欠费销号中主动离网占据了离网客户的绝大部分,被动销号比例极少。

欠费被动销号的主要原因是客户有多张卡或是由于工作原因暂时离开本地,对于不使用的号卡客户未办理停机保号业务或办理停机保号业务后预存话费不足,导致预存话费抵扣完后号卡欠费离网。

欠费主动销号主要原因是客户为了逃避欠费,而重新购卡以得到更多的优惠而选择主动弃卡离网。对低端客户而言,其离网重入网的成本是重新购买 SIM 卡的费用和变换手机号码后通知朋友所需要的费用或负效用;客户的收益包括:

(1) 新入网优惠话费,或者新旧资费套餐的优惠;

(2) 代理商的让利;

(3) 欠费滞纳金豁免收益;

(4) 平均单个客户欠费中未被追缴的部分。

因此,客户净收益是上述四部分收益减去客户重入网成本的余额。

10.5.6 应用

建立模型的最主要目的就是要应用到实际中,客户流失预测模型也是为了在实际工作中及时对在网客户未来的一种可能行为做出比较准确的预测,使得运营商能够及时地采取相应措施,挽留这部分可能流失的客户中有价值的群体,增强企业的竞争力,增加企业的收益。

1. 流失影响因素的分析

根据流失预测模型的输出结果,结合企业外部竞争环境的变化寻找影响客户流失的环境原因。通信企业从深入分析预测模型结果、收集企业外部竞争情报和对已流失客户一对一沟通等方面入手,将数据分析和环境分析相结合,找出造成客户流失的真正原因。

根据以上分析,得出影响客户流失因素的相对重要性,如图 10.14 所示。

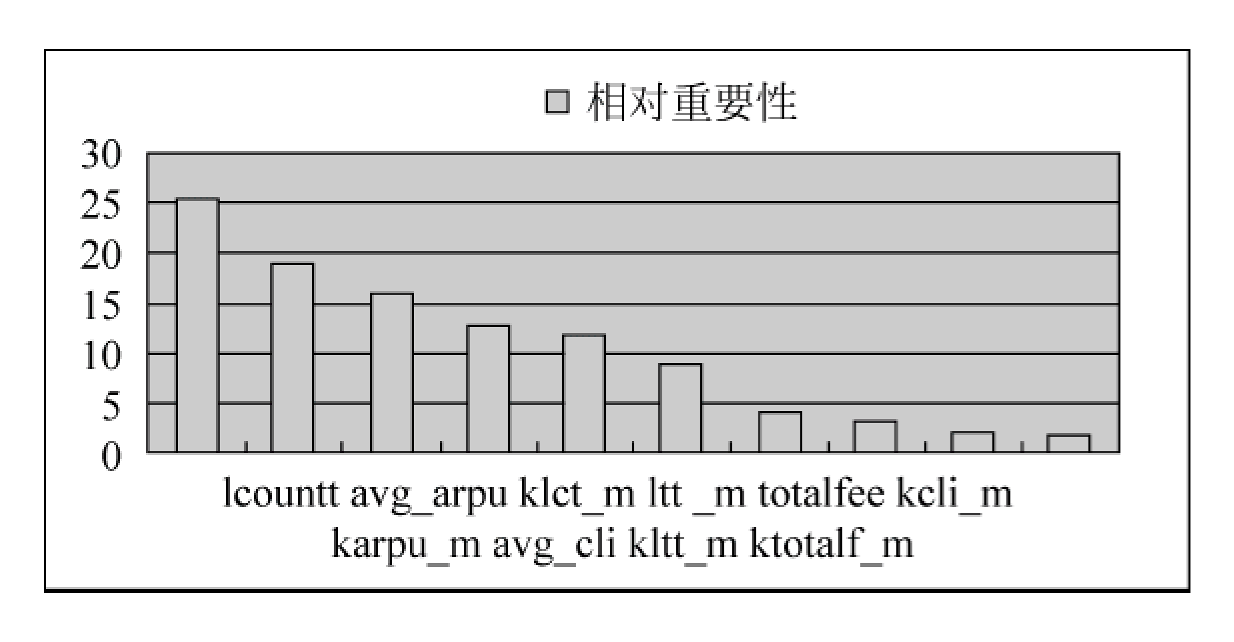


图 10.14 客户流失因素分析

(1) 本地通话总时间和客户的 ARPU 值是鉴别客户失倾向的两个最主要的指标,在客户流失的决策树模型中,通话时间短的客户的流失比例要显著高于通话时间长的客户,低 ARPU 值客户流失倾向高于 ARPU 值客户,因此这类流失群体中,例如学生,外来务工人员

离开本地是造成流失的主要因素;从消费行为上看漫游指标是客户流失的一个显著性指标,漫游时间长的外地客户是一类高流失倾向的客户群。

(2) 同一样本客户在不同的月份消费行为存在一定的差异,从决策树的分类效果看,离流失时间最近的流失客户消费行为更能体现流失特征。

(3) 决策树分类模型具有一定的时效性,不同月份流失样本建立的决策树模型存在显著的差异,因此进行客户流失的预测必须考虑时间效应。由于采用非等比例的随机抽样方式,决策树中的流失比例不是真正的客户流失比例。第一个原因是从建立的决策树模型来看,分类客户中最高的流失比例在12%左右,不能直接进行客户流失预测。第二个原因在建模中,数据不够完备,缺乏如客户的个人信息、投诉信息、缴费信息等相关数据。

2. 营销策略制定

营销活动要取得良好的效果,必须充分了解客户的心理,设计出适合的营销策略。针对客户流失原因和因素,可以归结为以下几点:

(1) 客户的基本属性,如性别、年龄、在网时间、职业爱好、籍贯、入网品牌与号码。注册服务等级(SLA)等,这些资料在客户登记入网或客户调查等过程中得到,并永久保存在客户资料数据库中。不同背景的客户有不同的社会行为特征和爱好,如职业影响收入,年龄影响产品购买类型等。

(2) 客户的通话情况,如通话类型、平均通话时间、通话对象及亲情号码、通话地点、漫游类型、数据业务使用情况、消费积分、客户价值类型、客户 VPMN 群等在计费中心客户消费话单和账单中可以获取客户消费的详细信息。

(3) 客户服务情况,指客户通过投诉渠道或客户服务界面(如营业厅、网站、客户经理等)进行的有关缴费、服务投诉/建议的情况。客户满意度高,投诉就少;查询月消费详细清单的客户对资费政策通常很敏感等。

(4) 网络运行质量情况,包括通信网络的质量,小区或无线信道通话接通率等影响客户消费的网络因素。网络维护部门通过对交换机进行话务测量可以获得网络运营的详细分析报告。网络质量是移动通信企业的生命,是客户服务与业务领先的基础。

(5) 资费政策与技术进步,不同运营商的资费高低改变等也是影响流失的一个重要因素。技术革新会带动一大批追新潮客户使用最新技术产品。

选择待挽留目标客户和赢回目标客户,制定客户挽留策略和客户赢回策略。市场部门针对这些名单,通过打关怀电话、设计优惠套餐、送小纪念品等多种方式来挽留流失倾向大的客户,最大限度地降低客户流失率。

3. 营销活动后评估

实施营销方案,并对方案的实施效果进行跟踪和评估。需要指出的是,在实施目标活动后,要根据市场环境的变化和客户行为的新特点修正客户流失预测模型。因此,整个客户流失分析流程应该是一个往复的循环过程。

10.6 WAP日志挖掘

中国通信业的快速发展令世界瞩目,截至2012年12月底,中国移动电话用户数达11.04亿。全球移动电话用户数约62亿,普及率达87%;中国互联网用户数达到5.64亿,

普及率达 42.1%。目前,中国的移动手机用户数和互联网用户数均居世界第一。随着 3G(the 3rd Generation)牌照的发放,3G 的到来加速了移动通信和互联网的融合,并呈现三个趋势,即互联网接入的移动化、移动业务的互联网化、互联网业务的移动化。

由于传统语音业务的价格持续下降,促使运营商开始转变收入重点,将数据业务逐步调整为未来利润的主要来源,数据业务被普遍认为是电信运营商的下一个金矿。而 WAP(Wireless Application Protocol)业务是当前网络环境下最重要的数据业务之一,几乎所有的手机终端均内置了 WAP 浏览器,使得大量成熟的数据业务以 WAP 作为重要的推广渠道,WAP 业务在运营商的业务架构中地位越来越重要,其用户数量也飞速发展。但是随着用户数量的快速增长,WAP 业务质量却没有跟上用户发展的脚步。WAP 业务同质化严重,SP(Service Provider,服务提供商)的注意力集中在如何从用户身上获得更多的话费,甚至利用 WAP 业务中的一些技术漏洞强行将用户绑定,引起大量的投诉。但随着运营商对 WAP 业务管理的逐步规范,对 WAP 用户的使用行为进行深入挖掘成为必然,并具有一定的现实意义。

10.6.1 定义

WAP 日志在一定程度上反映了 WAP 用户的使用行为习惯和特征,主要包括系统日志和用户访问日志。系统日志记录了 WAP 服务器在运行过程中系统的各种状态,为改进服务器性能、故障排除等提供了重要的参考依据,可以帮助系统维护人员快速定位故障并解决;用户访问日志记录了用户浏览 WAP 页面时的各种信息,包括用户访问时间、访问页面地址、访问机型参数、用户 IP 地址和用户标识等。

不同的系统会产生不同格式的 WAP 日志,根据系统性能需求系统管理员可以配置服务器产生日志的复杂度。在服务器空间有限且性能不高的情况下,可以缩减日志参数,仅记录与用户访问信息相关的内容。在服务器硬件条件允许的情况下,建议尽可能多地记录用户访问参数,因为在日志提取阶段很难客观判断日志参数在未来挖掘中的价值,尽可能多地保留以避免具有潜在价值的数据丢失。

这里的 WAP 日志是基于微软公司的 Internet Information Services 5.0 生成的,按照日期对日志进行命名,每天的用户访问日志存储在一个文本文件中,例如 20080601.txt。日志文件中各参数之间用空格分开,一行为一个用户的访问行为。

WAP 日志的具体参数名称和含义如表 10.30 所示。

表 10.30 WAP 日志参数名称和含义

参数名称	含义	功能	实例
Date	日期	用户访问 WAP 的日期	例如 2013-07-25
Time	时间	用户访问 WAP 的时间	例如 00:00:01
c-IP	用户访问 IP	访问用户的 IP 地址	例如 211.137.167.133
CS-username	访问用户名	访问 WAP 的用户名	
s-IP	服务器端 IP	用户访问 WAP 页面对应的服务器端地址	例如 211.157.8.68

续表

参数名称	含　　义	功　　能	实　　例
s-port	服务器端口	用户访问 WAP 页面对应的服务器端口	例如 8080
CS-method	请求方式	客户端操作请求的种类	例如 GET，POST
CS-URI-stem	请求内容名称	用户访问的 WAP 页面地址	例如/desk/cxzd/intro.asp
CS-URI-query	请求内容参数	用户访问 WAP 页面地址所带的参数	例如 MISC_ID=999&MISC_SessionID=999
sc-status	状态代码	用户访问 WAP 页面的状态代码	例如 200，302
CS-User-Agent	用户代理	用户访问 WAP 所使用的手机型号	例如 Nokia3108

10.6.2 数据准备

数据准备阶段的工作主要是选择合适的数据源，整理分析所需的 WAP 日志，对原始数据进行检查和预处理，生成新的衍生变量，并整理成适合分析使用的数据宽表。

数据准备过程分为如下几个步骤。

1. 选择数据分析范围

根据需求选取移动梦网某一 WAP 产品 2008 年 5～8 月连续四个月的用户访问日志作为分析对象。

2. 收集原始数据

WAP 产品日志是最原始文件，和互联网网站日志类似，其中记录了用户在浏览 WAP 页面过程中的大量信息。不同的服务器系统所产生的日志略有不同，大多都包括用户 IP 地址、访问终端类型、访问时间和所访问页面地址等，而移动梦网 WAP 产品同时还可以获得用户手机号码，这是区别于互联网日志的一个重要特点，即电信业务最大的特质——用户个性化标识。针对用户的个性化行为分析将紧密围绕着这一标识进行。

3. 数据清洗

日志一般存储于文本文件中，虽然大多都是按照一定规则记录的，但在今后的分析中通过文本文件方式进行信息的查找较为不便，故需要把日志内容导入到数据库中或者更好的存储介质中以备操作。此过程尽量避免将日志中含有的信息删掉，让日志中的信息尽量完整地在新存储介质中得到展现。

4. 检查数据质量

可以通过对所提取数据的时间分布进行直观观察，对异常时间点进行针对性分析；也可以观察所提取参数的数值分布，对其可靠性进行评估。由于用户使用 WAP 的行为具有较强的周期性，可以比对不同周期的相同时间点，对异常数据进行检验。

5. 计算中间变量

对 WAP 用户的使用行为数据进行加工处理，生成新的衍生变量。

6. 合并生成宽表

按照用户 ID 合并地域、终端、产品内容和用户四类指标，生成一张数据宽表，其中每行代表一个用户，每列代表一个变量，如表 10.31 所示。

表 10.31 数据宽表中各指标变量

名称		含义	功能	计算方法
地域类	地域名称	省份名称	标识分析结果所对应的省份	日志中用户参数的前四位标识用户所在区域,可通过对照区域编号表确定访问用户所在区域
	时间	数据产生的时间,以天为统计单位	反映产品运营过程中的时间变化趋势	日志中的时间参数
	累计浏览用户数	从产品投入运营至今访问过 WAP 产品的用户总数	反映产品投入运营后各地用户的规模	对日志中的用户标识进行排重,计算当天访问用户数,再与之前累计用户数相加
	浏览用户数	统计期内浏览过 WAP 产品的用户数	当日访问 WAP 产品的用户数	对日志中的用户标识进行排重,计算得到
	新增浏览用户数	统计期内浏览用户数较前一日增加的数量	衡量浏览 WAP 产品的用户数量变化情况,常用于异常检验	当日与前一日浏览用户数之差
	订阅用户数	有过订阅频道行为的用户数	反映用户订阅 WAP 业务的活跃程度(由于该 WAP 产品采用类似于 RSS 的订阅模式,在用户浏览内容前先要进行频道订阅)	用户订阅成功后,访问频道日志文件将记录用户所访问的频道参数,根据参数即可判断用户是否为订阅用户
	订购套餐用户数/收费用户数	订购收费套餐的用户数量	描述用户对价格敏感的程度,以及对收费频道的认可度(该 WAP 分为收费频道和免费频道两种,对于优质内容用户需要付费才能浏览)	计算日志文件中访问“订购成功”页面的用户数进而得到当天订购套餐用户数
	新增收费用户数	当日新增的收费用户数	描述收费用户数的变化情况,可了解 WAP 产品收入的变化趋势	计算当日收费用户数和前一日收费用户数之差得到
	主动浏览用户数	浏览详细内容的用户数	描述用户对每日更新内容的兴趣程度(该 WAP 产品采用先浏览内容摘要,用户对详细内容感兴趣再点击浏览完整的文章内容的模式)	提取日志中访问详细内容页面的用户行为记录
	主动浏览用户占比	主动浏览用户在浏览用户的占比	反映对 WAP 产品感兴趣的用户占整体用户群的比例	主动浏览用户数除以浏览用户数
	浏览用户占比	浏览用户在累计访问用户的占比	用户对 WAP 产品整体的认知度	浏览用户数除以累计访问用户数
	人均主动浏览次数	统计期内平均每个用户的主动浏览次数	用户对产品内容的兴趣程度	计算所有用户访问详细页面时产生的页面参数的总数,然后除以访问用户数
	人均订阅频道数	累计访问用户中平均每个用户订阅频道数量	用户对于 WAP 产品的频道内容的兴趣度	订阅用户数除以累计访问用户数
	添加频道用户数	统计期内添加频道的用户数	反映每日用户添加频道的活跃程度,针对此变量的研究可以得出用户心理的周期性规律	
	删除频道用户数	统计期内删除频道的用户数	反映每日用户删除频道的情况	

续表

名　称		含　义	功　能	计算方法
终端类	终端名称	数据对应的终端名称	标识分析结果对应的终端	用户访问 WAP 服务器时，日志会记录用户的 user-agent 即手机标识，简称 UA。利用 UA 与终端名称的对应表便可得到准确的终端型号
	时间	数据产生的时间，以天为统计单位	反映产品运营过程的变化趋势	日志中的时间参数
	累计浏览用户数	从产品投入运营至今，曾经通过终端访问 WAP 产品的用户总数	衡量产品投入运营后各终端用户的规模	对于不同型号终端的用户行为日志进行用户排重，得到当天访问用户数，再与之前累计用户数相加
	浏览用户数	统计期内通过终端浏览过 WAP 产品的用户数	当日访问 WAP 产品的用户数	对用户行为数据基于用户标识进行排重，可得到当天浏览用户数
	新增浏览用户数	统计期内通过终端浏览的用户数较前一日增加的数量	衡量各终端用户浏览产品的变化情况，常用于趋势分析以及异常检验	计算某终端的当日与前一日浏览用户数之差
	订阅用户数	某终端有过订阅频道行为的用户数	各终端用户订阅频道的活跃程度	用户订阅成功访问频道时，日志将记录用户所访问的频道参数，根据参数即可判断用户是否为订阅用户
	订购套餐用户数/收费用户数	某终端订购收费套餐的用户数	各终端对价格敏感的程度	以 UA 为标识计算日志中访问了"订购成功"页面的用户数，可得到当天某终端订购套餐用户数
	新增收费用户数	统计期内某终端新增的收费用户数	某终端收费用户数的变化情况	某终端的当日收费用户数减去前一日收费用户数
	主动浏览用户数	某终端用户浏览详细内容的数量	各终端用户对每日更新内容的兴趣程度	提取日志中某终端访问详细内容页面的行为记录
	主动浏览用户占比	某终端主动浏览用户在浏览用户的占比	反映某终端对 WAP 产品感兴趣的用户占整体用户群的比例	主动浏览用户数除以浏览用户数
	浏览用户占比	某终端浏览用户数在累计访问用户的占比	某终端用户对 WAP 产品整体的认知度	浏览用户数除以累计访问用户数
	人均主动浏览次数	统计期内某终端平均每个用户的主动浏览次数	用户对产品内容的兴趣程度	计算某终端的所有用户访问详细页面产生的页面参数的总数，然后除以访问用户数
	人均订阅频道数	某终端累计访问用户中平均每个用户订阅频道数	各终端用户对于 WAP 产品的频道内容的兴趣度	订阅用户数除以累计访问用户数
	添加频道用户数	统计期内某终端添加频道的用户数	每日用户添加频道的活跃程度	通过日志文件对添加频道成功页面的访问记录，以终端型号、用户手机号为标识，计算添加频道用户的总数
	删除频道用户数	统计期内某终端删除频道的用户数	某终端用户删除频道的情况	

续表

名　称		含　义	功　能	计算方法
产品内容类	时间	数据产生的时间,以天为统计单位	反应频道数据的变化趋势	日志中的时间参数
	频道名称	数据对应的频道名称	标识分析结果对应的频道	用户访问 WAP 服务器时,日志会记录用户所访问的频道参数,利用参数与频道名称的对应表可知用户访问的频道名称
	频道类型	频道的分类	频道类型分为初始频道和可选频道(该 WAP 产品对免费用户默认设置三个初始频道,便于用户体验)	先利用用户访问记录得出所访问频道的参数,再利用频道参数对应表可知访问频道类型
	计费方式	收费/免费方式计费	频道的收费方式	先利用用户访问记录得出所访问的频道参数,再利用频道参数对应表可知频道计费方式
	订阅用户数	订阅某频道的用户总数	用户订阅频道的活跃程度	分析每日用户访问"订阅成功"页面的日志,通过频道参数对应表可知用户新订阅的频道名称,再将新的订阅信息插入到用户频道订阅表中存储
	新增订阅用户数	统计期内新增的订阅用户数	反映各频道订阅用户数的变化情况,便于对频道发展情况做出相应的预测	统计周期内对用户访问"订阅成功"页面的所产生的日志文件进行分析,通过频道参数对应表找出用户新订阅的频道名称
	退订用户数	统计期内,用户取消订阅某频道的用户数	反映用户对哪些频道内容不满意	对用户访问"删除频道成功"页面产生的日志进行分析,通过频道参数对应表得到用户退订频道名称,并插入到用户订阅/退订频道历史数据库
	主动浏览用户数	统计期内浏览某频道详细内容的用户总数	用户对内容的兴趣度	利用用户访问详细内容页面时日志记录的频道参数,通过频道参数对应表得到所要统计的频道数据,再结合用户标识得出主动浏览用户数
	浏览用户数	统计期内浏览某频道的用户数	用户对频道的兴趣程度	利用用户访问频道页面时日志记录的频道参数,通过频道参数对应表得到所要统计的频道名称,再结合用户标识得到浏览用户总数
	主动浏览用户占比	统计期内某频道浏览详细页面的用户数在所有访问该频道用户数的占比	频道内容对用户的吸引程度	主动浏览用户数除以浏览用户数
	人均主动浏览次数	统计期内平均每个用户浏览该频道详细内容的次数	反映频道的粘性	利用日志记录的访问频道参数,算出访问频道页面的次数,除以主动浏览用户数

续表

名　　称		含　　义	功　　能	计算方法
用户类	用户 ID	唯一标识用户的字符串	唯一标识用户	日志中记录访问用户的用户 ID 标识
	城市名称	用户所在城市		根据日志文件中记录的用户访问 IP 地址，与各地市 IP 地址表进行比照得到用户使用业务所在地
	终端型号	用户使用终端的型号		用户访问 WAP 服务器时，日志会记录用户的 user-agent（即手机标识，简称 UA），利用 UA 与终端名称的对应表得到准确的终端型号
	频道订购情况	用户是否订阅该频道（包括 22 列，每列为一个频道）	反映当前用户群订阅频道的分布情况	通过用户订阅频道历史数据表得到

10.6.3　建模过程

下面介绍 WAP 日志挖掘的建模过程。

1. 数据整合

由于数据仓库中地区、终端、频道等数据相对独立，所以需要先将各部分数据进行整合，最终得出完整的用户订阅频道信息表。先利用用户订阅信息表中的 user_id、city_id、region_id、mobile_type 等字段进行关联，把之前可读性较差的数字表示方式替换为有具体直观意义的字符串方式（在进行挖掘过程中就不需要再到历史数据表里查找相关参数），然后利用填充结点将表中的“null”值替换成“F”，这样订阅频道的相关信息就以布尔值“F”和“T”表示。最后利用表格结点对处理结果进行检查，提高数据质量。

2. 聚类

因为 TwoStep 聚类算法可以自己根据数据的分布进行类数的选择，所以先利用 TwoStep，并将类别数作为后继 k-Means 聚类的输入参数，以获得数量较平均，类间距离较大的最佳聚类结果。

3. 结果展示

网络图可以直观显示聚类结果，例如订阅娱乐频道的用户中有很多同时订阅了焦点评论，在网络图中两个频道之间会形成一条很粗的线，以表示两者之间呈较强的相关性。各点之间的线段即表示两个频道的关系，线段越粗相关性越大，越细相关性越小。网络图提供了动态的显示方式，分析人员可以通过调节可拖动的进度条进行条件的改变，以便将关联度强（线条越粗）的各组频道从繁杂的关系中提取出来。

4. 解释和评价

首先对聚类结果进行直观观察，通过分布结点了解聚类的用户群，然后将聚类结果作为

输出变量,地区、终端类型、各频道的订阅情况作为输入变量,利用决策树进行聚类结果的解析,从而以逻辑表达式解释聚类结果。

10.6.4 结果

根据上述分析过程,结果如表10.32所示。

表10.32 WAP日志聚类结果

序号	类别名称	订购行为特点	营销方案
1	证券型	此类用户对证券、财经极其关注	可以定向推送证券相关的增值信息(例如手机证券报),也可将与证券相关的内容进行打包,作为提升WAP业务订购的一种手段。关注证券相关内容的用户往往是非常活跃的用户,各种信息的获取需求很大,是点击次数产生的主要用户群。基于此用户群特质,可通过WAP内嵌广告的方式转化为收入,达到最佳的盈利模式
2	娱乐型	此类用户对星座、动漫非常关注	通过此类用户的订购情况看,可以判断此用户群以年轻学生群体为主,接受新鲜事物很快,对价格敏感,有较强的发表观点的意愿,渴望自我实现。可以增加评论等社区功能,为用户主动推荐“娱乐”、“时尚”等年轻人可能感兴趣的频道,充分迎合年轻群体的需求
3	体育型、男性化	此类用户对体育、军事很感兴趣	通过此类用户的订阅情况看,可以判断用户群具有男性化的特质。对于体育方面的内容可以进行领域细分(如体育可分为篮球、足球等),提高用户获取信息的准确性。也可引入WAP赛事直播的方法,让对比赛感兴趣但无法收看电视转播的人群,通过文字的方式实时了解赛事情况,从而提高产品的黏性。对于军事方面的内容,可以增加时事政治方面的内容推荐,让用户有一个更加全面的军事视角。根据男性特质该用户群还可以提供“创业”、“女性”等相关内容,以开辟吸引此类用户的新内容领域
4	成熟型	此类用户对军事、焦点评论和证券很感兴趣	此用户群所感兴趣的内容有较强的深度,可归于成熟型用户。对这类用户最重要的就是所选内容的价值,若能够满足其对内容深度的需求,单个用户的ARPU值会较高

通过网络图可以了解WAP产品整体用户类型情况,在营销资源有限而导致不能针对分类用户群进行精确化营销的前提下,可先对整体用户群进行营销,大大提高了运营分析转化为实际运营的可行性。

每个频道在网络图中均以一个点表示,而点之间的线段表示频道间的关联程度。图10.15将所有的关系进行了呈现,关系看起来很复杂,可以通过调节阈值进行过滤,以发现强关联。

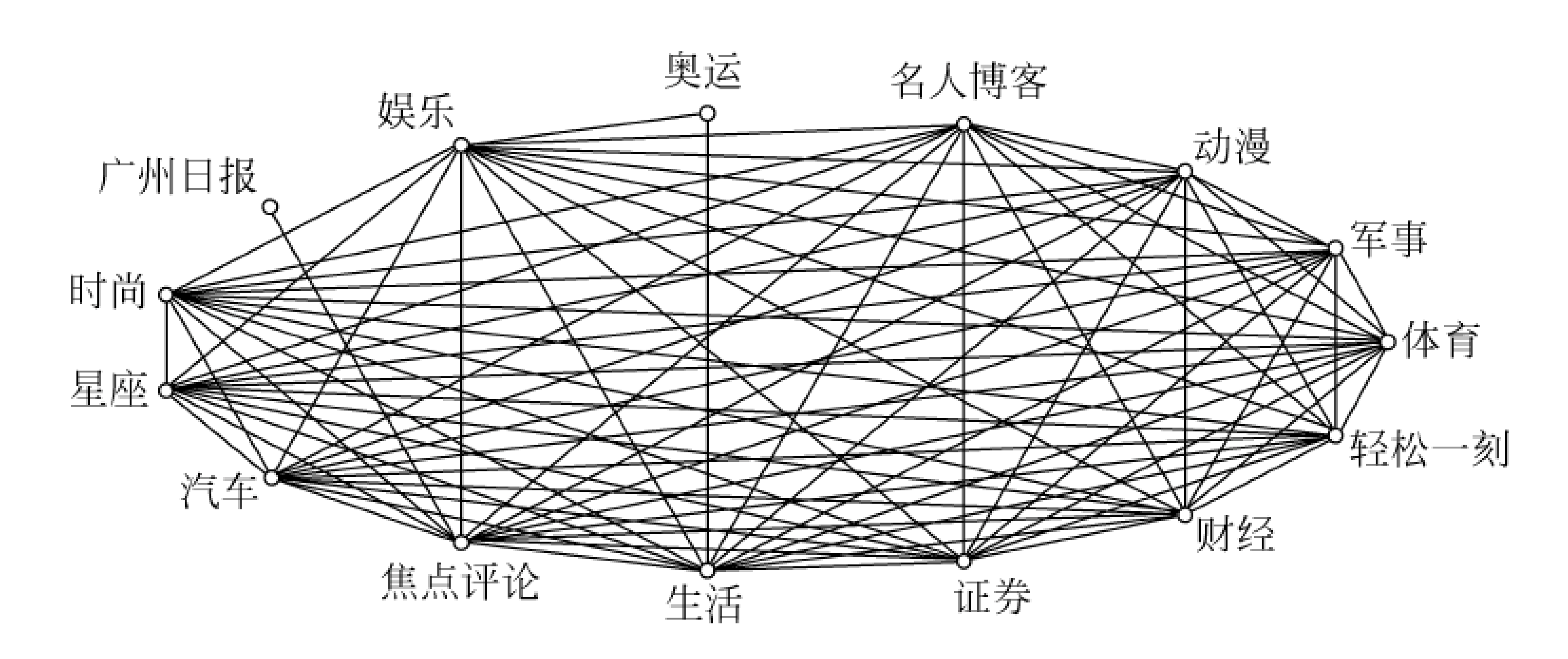


图 10.15　初始的网络图

设定合适的阈值，形成如图 10.16 所示的关系图。由图 10.16 可知，该 WAP 业务订阅最多的四个频道分别是娱乐、轻松一刻、生活和焦点评论，并且各个频道之间都存在较强的关联，形成了 WAP 业务用户群频道订阅的基础组合。

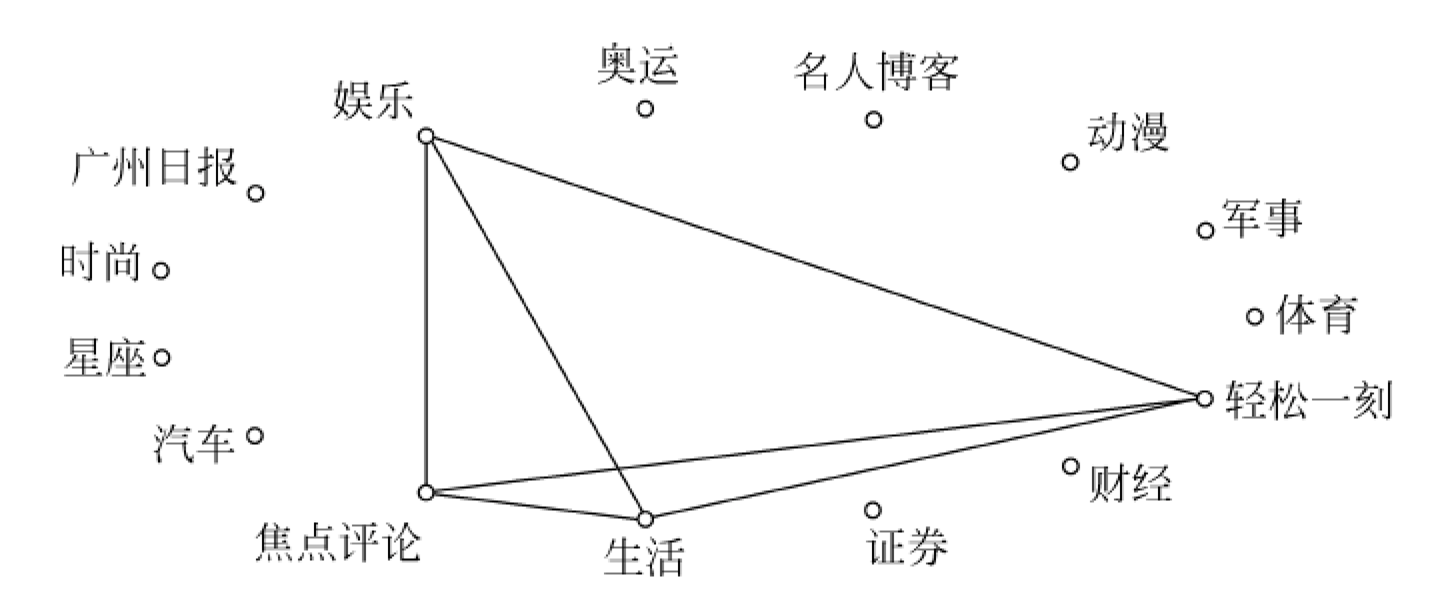


图 10.16　过滤后的网络图

第三篇

语义网和本体

第11章　知识基础
第12章　语义网和本体

第 11 章　知 识 基 础

知识是人们日常生活及社会活动中常用的术语之一。例如人们常说“知识就是力量”，“应该多学点知识”等等。什么是知识？知识有哪些特点？它与平常所说的信息有什么区别？如何表示知识？知识表示与人工智能、知识工程的关系如何？针对上述问题，本章将从知识的定义、分类、度量和表示等几方面加以阐述。

11.1　概述

1. 数据

人类赖以生存的空间是一个物质的世界，同时又是一个信息的世界。在这个不断变化的世界中，无论是政治、经济、军事，还是科技、文化、教育等，时时刻刻都在产生大量的信息。谁能及时地掌握有用信息，并能把有关信息关联起来加以充分利用，谁就能在激烈的竞争中立于不败之地。随着社会的发展和进步，信息在人类生活中越来越扮演着极其重要的角色。但是，信息需要用一定的形式表示出来才能被记载和传递，尤其是使用计算机进行信息的存储和处理时，更需要用一组符号及其组合进行表示。像这样用一组符号及其组合表示的信息称为数据。

由此可见，这里所说的数据已不仅仅是通常意义下的“数”，而是在概念上的延伸，泛指对客观事物的数量、属性、位置及其相互关系的抽象表示。它既可以是一个数，如整数、小数、正数和负数，也可以是由一组符号组成的字符串，如姓名、性别、地址或消息等。

数据和信息是两个密切相关的概念。数据是信息的载体和表示，信息是数据在特定场合下的具体含义，换言之，信息是数据的语义，只有把两者密切地结合起来，才能实现对现实世界中某一具体事物的描述。另外，数据和信息又是两个不同的概念。对于同一个数据，在某一场合下可能表示一个信息，但在另一场合下却可能表示另一个信息。例如数字“6”是一个数据，既可以表示“6 本书”、“6 支铅笔”，也可以表示“6 个人”或“6 部手机”等。同样，对同一个信息，在不同场合下也可用不同的数据表示，如同对于同样的一句话，不同的人会用不同的言语来表达一样。

2. 信息

20 世纪 40 年代末，美国数学家香农提出信息熵的数学公式，从量的方面描述了信息传输和提取问题，创立了信息论。首先，信息论在通信工程领域得到广泛应用，为信息科学的研究奠定了初步基础。

在消息传递系统中所传输的是消息，但消息传递过程中，最普遍却容易被忽视的一点是接收者在收到消息之前不知道消息的具体内容。对于接收者而言，消息的传递过程是一个从不知到知的过程，或者说是一个从不确定到确定的过程。

从通信过程来看，接收者的所谓不知就是不知道发送端将发送描述何种运动状态的消息。例如看天气预报前，并不清楚天气将如何；看天气预报后，这种不确定性大大减小。不确定性消除了，接收者就获得了信息。因此香农认为，信息是被消除的不确定性。

关于信息的定义,后来的学者又有种种讨论。由于信息概念的复杂性,在定义信息时必须十分注意定义的条件,应当根据不同的条件区分不同的层次给出信息的定义。最高的层次是普遍的层次,也是无条件约束的层次,我们称之为本体论层次。在这个层次上定义的信息是最广义的信息,其适用最广。然后,如果引入一个条件加以约束,则最高层次的定义就变为次高层次的定义,次高层次的信息定义的适用范围比最高层次的定义的适用范围要窄,所引入的约束条件越多,定义的层次越低,所定义信息的适用范围就越窄。这样,根据引入条件的不同,可以给出不同层次和不同适用范围的信息定义,这些不同的信息定义构成了信息定义的体系,即:

- 本体论层次的信息是事物运动的状态和状态改变的方式。
- 认识论层次的信息是认识主体所感知或所表述的事物运动的状态和方式。
- 语法信息是认识主体所感知或所表述的事物运动状态和方式的形式化关系。
- 语义信息是认识主体所感知或所表述的事物运动状态和方式的逻辑含义。
- 语用信息是认识主体所感知或所表述的事物运动状态和方式相对于某种目的的效用。
- 先验信息是指观察者在观察某事物之前通过某种途径所感知的该事物运动状态和方式。
- 实得信息是指在观察过程中,观察者通过观察所新感知到的该事物运动状态和方式。
- 实在信息是指该事物实际的运动状态和方式,这也是在理想观察条件下观察者所获得的关于该事物的全部信息。

上述语法信息、语义信息、语用信息、先验信息、实得信息和实在信息都是认识论层次上的各种信息概念。进一步,如果不仅对观察者施加各种限制条件,而且对所观察的事物也规定一些限制性约束,则会得到层次更低、适用范围更小的信息定义。例如,如果限定所观察事物的运动方式是随机型的,则可以分别得出概率性的实在语法信息、概率性的先验语法信息、概率性的实得语法信息等等;如果限定所观察事物的运动方式是半随机型的,则可以分别得到偶发性的实在语法信息、偶发性的先验语法信息、偶发性的实得语法信息等等;如果限定所观察事物的运动方式是确定性的而运动状态是模糊的,则可以分别得到模糊实在语法信息、模糊先验语法信息、模糊实得语法信息等等。类似的情形也适用于语义和语用信息。总之,对观察者、观察对象(即事物)以及观察过程的性质都可以规定各种不同的条件,因此可以得到层次高低不同、适用范围各异的各种信息定义。

3. 知识

从古希腊开始,人类从未停止对于知识的研究与探索。哲学家研究有关知识的一般特性与规律,而自然科学家孜孜不倦地获取具体的知识。20 世纪中后期,这种研究格局发生了变化。由于知识在人类文明中发挥的作用越来越大,不仅是哲学家、逻辑学家、教育学家和心理学家,而且计算机科学家都在认真地研究知识的一般特性与规律。这是因为人类已经进入了信息化社会,而且正在向知识化社会迈进。

古希腊哲学家苏格拉底认为知识的唯一功能是自我认识,即人的智力、道德和精神的生活成长。毕达格拉斯认为知识的目的是通过使用知识的人知道他想说什么和怎么说,从而使其行为更有效,知识就是指逻辑、语法和修辞;中国儒家则认为知识是知道说什

么、怎么说以及出人头地和俗世成功的途径。对于道家而言,知识是自我认识和通向领悟智慧的途径。

知识工程的创始人 Feigenbaum 教授曾经说过:“知识和信息不一样,知识是信息经过加工整理、解释、挑选和改造形成的”。可以说,这是从广义信息论的角度对知识进行定义,相关的定义还包括(表 11.1 中给出了各种知识的定义,以下保留英文以便于对照):

▲ Knowledge can be considered as the distillation of information that has been collected, classified, organized, integrated, abstracted and value added. Knowledge is at a level of abstraction higher than the data, and information on which it is based and can be used to deduce new information and new knowledge. When considering knowledge it is usually in the context of human expertise used in solving problems.

▲ Organized body of information. The acquaintance with facts, truths or principles as from study or investigation or the familiarity with a partaker subject, branch of learning etc.

综上所述,信息在人类生活中占据着相当重要的地位。但是,只有把有关的信息关联在一起时才具有实际意义。一般地,把有关信息关联在一起形成的信息结构称为知识。一些具有代表性的知识定义如表 11.1 所示。

表 11.1 代表性的知识定义

序号	知 识 定 义
1	知识是通过实践、研究、联系或调查获得的关于事物的事实和状态的认识
2	知识是人们在长期的生活及社会实践中、科学研究及实验中积累起来的对客观世界的认识与经验,人们把实践中获得的信息关联在一起,就获得了知识
3	知识是人类在实践的基础上产生又经过实践检验的对客观实际的可靠的反映
4	知识是对科学、艺术或技术的理解,是人类获得的关于真理和原理的认识的总和
5	把有关信息关联在一起所形成的信息结构称为知识
6	知识是人脑创新的成果,是人类智慧的结晶。智慧是人类文明的源泉,是推动历史发展的永恒动力,是生产力诸要素中的核心

知识是人们在长期的生活及社会实践、科学研究及实验中积累起来的对客观世界的认识与经验,人们把实践中获得的信息关联在一起,就获得了知识。信息之间有多种关联形式,其中用得最多的一种是“如果,则”表示的关联形式,反映了信息间的某种因果关系。例如,我国的北方人经过多年的观察发现,每当冬季来临时,就会看到一批批的大雁向南方飞去,于是把“大雁南飞”与“冬天将要来临”这两个信息关联在一起,就得到了“如果大雁向南飞,则冬天将要来临”这样一条知识。

知识反映了客观世界中事物之间的关系,不同事物或者相同事物间的不同关系形成了不同的知识。例如“雪是白色的”是一条知识,反映了“雪”与“颜色”之间的一种关系;“如果头痛且流涕,则有可能患感冒”是一条知识,反映了“头痛且流涕”与“可能患感冒”之间的一种因果关系。

4. 信息和知识的关系

马克思曾说过:“任何科学只有在具有数学基础以后,才能算得上是真正的科学。(大意)。”我们首先从数学的角度区分信息和知识这两个概念。

香农曾经对信息的数学本质进行研究,提出并回答了从数学的观点看,信息是什么的问题。他认为信息是一个数学量,用来消除不确定性。这种不确定性可以用具有概率意义的熵度量。由此,概率论成了研究信息论的基本数学工具。

如前所述,知识是结构化的信息,或者说知识是用于消除信息的无结构性的。在这一观点中,知识的数学基础应该和信息的不同,其核心不是概率论,应该是描述结构的某种数学模型。寻求并分析这种结构,有望建立知识的数学描述。目前,国际上关于知识本体的研究愈来愈热,证实了知识结构性是一个根本问题的观点是有道理的,本体就是知识结构性的基本描述,这一点已经成为国内外有关专家的共识。

5. 表示与知识表示

表示是使用人造的体系(典型的例子是数学)对自然界事物的运动规律进行概括和抽象的模型,而这一模型可以预言自然界这种运动的所有情况。一旦这样的表示被找到,人们就认为这是对这类运动规律更深刻的认识。相比于自然的表示方法,它具有抽象性、深刻性和简洁性。对这种抽象意义上的表示,自然界中的物体是否可被感知,对表示没有什么直接的关系。换言之,表示与自然现象之间的形态上可以没有任何相同之处,它是自然现象在人为体系(公理)下的一种解释。

从传统意义上说,知识表示(Knowledge Representation)是概括智能行为的模型,属于人工智能的范畴,其特点是:

- 智能行为所特有的灵活性问题(常识问题)不能概括为一类简洁的理论,它是大量小理论的集合。
- 人工智能受到计算装置的约束。这就导致所采用的表示必须同时满足"刻画智能现象"与"计算装置可接受"这两个有时是矛盾的条件。正是对这两个条件的不同侧重导致了对表示的不同认识,并由此产生研究上的不同方法论。

可以说,知识表示是众多理论与技术的交叉学科,主要源于:

- 逻辑提供推理规则和形式化结构。
- 本体定义应用领域的各种存在。
- 计算支持知识表示从哲学到应用层面的实现。

如果没有逻辑,知识表示不可能明确,无法判别陈述是否矛盾或者冗余;如果没有本体,词汇、符号等就不能被很好地定义和使用,概念就不会具有很好的一致性;如果缺乏可计算的模型,逻辑和本体就不能由计算机程序实现,就不能获得具体应用。因此可以说知识表示是逻辑和本体为了实现某些领域特定应用的任务而建立的计算模型。

6. 人工智能与知识工程

人工智能主要研究采用人工的方法和技术模仿、延伸和扩展人的智能,实现机器智能。有人把人工智能划分为两大类:一类是符号智能,另一类是计算智能。符号智能是以知识为基础,通过推理进行问题求解,即所谓传统的人工智能;计算智能是以数据为基础,通过训练建立联系进行问题求解,如人工神经网络、遗传算法、模糊系统和人工生命等都可以包括在计算智能的范畴。

传统的人工智能主要运用知识进行问题求解。从实用观点看,人工智能是一门知识工程学,即以知识为对象,研究知识表示方法、知识运用和知识获取。

自1956年人工智能提出以来,已经取得了很大的进展和成功。1976年Newell和

Simon 提出了物理符号系统假设，认为物理符号系统是表现智能行为的必要和充分条件。这样，可以把任何信息加工系统看作一个具体的物理系统，如人的神经系统、计算机的构造系统等。20 世纪 80 年代 Newell 等人又致力于 SOAR 系统的研究，该系统是以知识块(Chunking)理论为基础，利用基于规则的记忆，获取搜索控制知识和操作符，实现通用问题求解。Minsky 从心理学的角度出发，认为人们在日常的认识活动中，使用了大批从以前的经验中获取并经过整理的知识，该知识是以一种类似框架的结构存在于人脑中。因此，20 世纪 70 年代他提出了框架知识表示方法。到了 20 世纪 80 年代，Minsky 认为人的智能根本不存在统一的理论。1985 年，他在自己发表的著作中指出思维社会是由大量具有某种思维能力的单元组成的复杂社会。以 McCarthy 和 Nilsson 等为代表，主张用逻辑来研究人工智能，即用形式化的方法描述客观世界。逻辑学派在人工智能研究中，强调的是概念化知识表示、模型论语义和演绎推理等。McCarthy 主张任何事物都可以用统一的逻辑框架表示，在常识推理中以非单调逻辑为中心。传统的人工智能研究思路是"自上而下"，其目标是让机器模仿人，认为人脑的思维活动可以通过一些公式和规则定义，因此希望通过把人类的思维方式翻译成程序语言输入机器，使机器有朝一日产生像人类一样的思维能力。这一理论主导了早期的人工智能研究。

1977 年，第五届国际人工智能联合会议上，美国斯坦福大学计算机系 Feigenbaum 教授做了关于"人工智能的艺术"(The Art of Artificial Intelligence)的讲演，提出知识工程这一名词，指出"知识工程是应用人工智能的原理与方法，对那些需要专家知识才能解决的应用难题提供求解的手段，恰当地运用专家知识的获取、表达和推理过程的构成与解释，是设计基于知识的系统的重要技术问题"。

从时间上划分，知识工程的发展大体经历了三个时期：

(1) 1965—1974 年为实验性系统时期。1965 年 Feigenbaum 教授与其他科学家合作，研制出 DENDRAL 专家系统。这是一种推断分子结构的计算机程序，该系统存储了非常丰富的化学知识，其解决问题的能力达到专家水平，甚至在某些方面超过同行专家的能力，其中包括它的设计者。DENDRAL 标志着专家系统的诞生。

(2) 1975—1980 年为 MYCIN 时期。20 世纪 70 年代中期 MYCIN 专家系统研制成功，它是一种用医学诊断治疗感染性疾病的计算机程序"专家系统"。MYCIN 是规范性计算机专家系统的代表，许多其他专家系统都是在 MYCIN 的基础上研制而成的。MYCIN 不但具有较高的性能，而且具有解释和知识获取功能，可以用英语与用户对话，回答用户提出的问题，还可以在专家指导下学习医疗知识，该系统还使用了知识库的概念和不精确推理技术。MYCIN 对计算机专家系统的理论和实践都具有较大的贡献。

(3) 1980 年以来为知识工程的产品在产业部门开始应用的时期。人工智能的研究表明，专家之所以成为专家，主要在于他们拥有大量的专门知识，特别是长期从实践中总结和积累的经验技能知识。从知识工程的发展历史可以看出，知识工程是伴随着"专家系统"的研究而产生的。实际上，知识工程的焦点就是知识。知识工程领域的主要研究方向包含知识获取、知识表示和推理方法等，其研究目标是挖掘和抽取人类知识，用一定的形式表示知识，使之成为计算机可操作的对象，从而使计算机具有一定的人类智能。

11.2 知识分类

随着人类对于知识内涵认识的深入,从不同角度对知识进行了分类。就某种意义而言,知识分类恰恰是建立在对知识内涵的理解之上的,分类原则本身也在一定程度上体现出人类在不同社会经济形态下对知识作用的不同认识。

德国哲学家马克斯·舍勒将知识划分为应用知识、学术知识和精神知识三大类。在此基础上,著名美籍经济学家弗里兹·马克卢普在20世纪中叶提出了知识产业理论,按照认识者的主观解释分析知识的种类,认为知识包括五个方面的内容,即实用知识、学术知识、闲谈和消遣知识、精神知识和不需要的知识(多余的知识)。另外,马克卢普还从科学的与历史的、一般抽象的与特殊具体的、分析的与经验的、永恒的与暂时的角度,对知识类别进行了概要分析。随后,马克卢普又从世俗知识、科学知识、人文知识、社会科学知识、艺术知识、没有文字的知识(如视听艺术)等角度对知识进行解释,提出知识具有真实、美丽和优秀等性质。

随着知识经济理论的逐渐发展,经合组织(Organization of Economic Cooperation and Development,OECD)对知识的分类成为目前最具权威和流行的一种。根据该组织的划分标准,将知识归纳为四种类型,即事实知识(Know-what)、原理知识(Know-why)、技能知识(Know-how)和人力知识(Know-who)。以上是从知识使用的角度划分的,因而更注重知识的实践性和价值性。为了更深刻地理解知识的含义并对其进行有效的管理,在OECD分类的基础上,进一步将知识划分为显性知识和隐性知识两大类。

所谓显性知识,是指可以通过正常的语言方式传播的知识,典型的显性知识主要是指以专利、科学发明和特殊技术等形式存在的知识,存储在书本、计算机数据库、CD ROM中。显性知识是可以表达的、有物质载体的和可确知的。在OECD所划分的四类知识中,关于Know-what和Know-why的知识基本属于显性知识;所谓隐性知识或称为隐含经验类知识(Tacit Knowledge),往往是个人或组织经过长期积累而拥有的知识,通常不易用语言表达,也不可能传播给别人或传播起来非常困难。例如技术高超的厨师或艺术家可能达到世界水平,却很难将自己的技术或技巧表达出来从而将其传播给别人或共享。隐性知识对应的是OECD分类中Know-how和Know-who的知识,其特点是不易被认识到、不易衡量其价值、不易被其他人所理解和掌握。

显性知识和隐性知识的划分突破了过去人们对于知识的认识,将还未经系统化处理的经验类知识予以承认。如果把显性知识比喻为“冰山的尖端”,则隐性知识就是隐藏在水面下的大部分,虽然它们比显性知识难发觉,却是社会财富的最主要源泉。知识管理中的一个重要观点就是隐性知识比显性知识更完善、更能创造价值,隐性知识的挖掘和利用能力,将成为个人和组织成功的关键。

11.3 知识表示

知识表示的最基本作用是能够清晰明确地表示面向计算机的知识。此外,知识表示还具有以下作用。

1. 突显问题本质

计算机在表示事物时，为保持知识表示的紧凑性和一致性，要求能抓住事物的本质和相互之间的重要区别，避免表示不必要或不可能知道的细节。所以合理的知识表示形式能突显问题的本质。

2. 支持知识获取

人工智能只有不断进化，才能突显其旺盛的生命力。所以知识表示必须能支持其渐增地从外界获取知识，使计算机内部模型越来越精确地反映外部世界，更好地完成问题求解任务。

3. 支持对知识库的高效搜索

如果计算机不但能够感觉到周围环境存在的问题，还能准确知道利用自己拥有的知识解决问题，这对其智能行为的产生将具有重要作用。所以知识表示应能支持对知识库的高效搜索，以便发现被感知的事物之间的关系和变化，找到对问题状态的最佳描述，消除重复、冗余的内容，处理感知信息中的错误。

知识表示研究的主要内容包括：

- 知识表示观的研究
- 知识表示方法的研究

针对知识表示观和知识表示方法，下面将分别介绍。

11.3.1 知识表示观

在讨论具体的知识表示方法之前，搞清楚“什么是表示”这一基本问题是十分必要的。根据对这个基本问题的不同理解和所采用的方法论，人工智能学界形成了不同的学派，对表示而言，可以称之为表示观。

1. 认识论表示观

认识论表示观认为对智能行为的刻画是与常识知识形式化紧密相关，因此对常识形式化的研究是人工智能的核心任务。常识推理在某种程度上就是问题求解中的灵活性，而灵活性的共同特点是不完全性、不一致性、不确定性和进化性，这些最终将与常识推理的可废弃性相联系。常识可以被说明在自然世界中的那些“什么均可以发生，什么也可以不发生”的现象。非单调推理是认识论学派研究的主流，而对“灵活性”的不同考虑与侧重产生了对常识研究的不同理论。

基于认识论表示观的主要特点是：

(1) 知识论表示观是在特定环境下对世界观察的结果，其意义在于说明表示是自然现象的一种替代形式。对人工智能研究而言，认识论表示观更加强调自然现象与表示之间的因果关系，即如果一种表示不能刻画某种智能行为，则失去了在 AI 范畴内研究的意义，而不管其形式是如何优美。这与物理学家的思考方法十分类似，但与数学家的完全不同。

(2) 认识论表示观认为启发式方法不属于表示的研究内容，其理由是对自然现象的表示是对这种现象的机制更深刻的刻画，至于怎样有效地得到行为描述与最后的合法结论不是认识世界的问题，而仅仅是怎样做得更好的问题。由于表示是对自然世界的刻画，因此从事实出发而推出结论的过程是合法的。另外，这种表示观对在计算机中有效地存储的考虑

并不是针对某些特定的已有表示方法,而是指由于常识知识的特点在于其存在着例外,因此需要有理论的概括才可有效地在计算机中存储它们。

综上所述,认识论表示观认为对常识知识的形式化是重要的任务,其含意不是指在AI中经常使用的穷举式的方法,而是寻找一种简洁地表示智能行为的理论。因此,这一表示观的要点就可根据P. Hayes的解释说明为"表示的唯一作用就是携带知识",这意味着表示可以独立于知识,当这个携带者中的变元被自然世界中的事实所代替时,知识将表现在其行为之中。

2. 本体论表示观

本体论表示观认为表示是对自然世界的描述,绝对的逼真是不可能的,自然世界唯一绝对精确的表示是其自身,其他表示都不是绝对逼真,任何表示不可避免地包含着简化或人为的规定。基于这样的考虑,产生了一系列的问题,这些问题的解决应当基于本体论表示观,即:

(1) 由于任何一种表示都是对自然世界事物的近似。因此,表示必然需要对世界的某个部分给予特别的注意(聚焦),而忽略世界的另外部分(衰减),而聚焦什么和衰减什么的"聚焦-衰减"效果(心理学称这种现象为注意力集中)就是看待外部世界的规定,这形成了本体论约定的集合。本体论约定必然性的理由是表示模型的不完善,而其有效性的理由则是因为注意力集中于世界的一小部分而达到对问题的有效求解。

(2) 本体论表示观强调对自然世界可以采用不同的方法记述,但注重的不是语言形式,而是内容,这与认识论表示观"表示的唯一功能是携带知识"的观点针锋相对。但本体论表示观又与知识工程表示观不同,它所注重的"内容"不是某些特定领域的特殊的专家知识,而是自然世界中的那些具有普遍意义的一般知识(General Knowledge)。寻找并建立这样一个具有常识知识并可为大多数领域使用的一般性知识库,就是本体论表示观中关于"内容"的含义。

(3) 本体论表示观认为,表示只是表述智能行为的部分理论,其暗示不考虑推理的纯粹表示是不存在的。这个观点与认识论表示观没有什么本质区别,区别在于表示的研究是否认为保真推理是其唯一需要遵循的原则。本体论表示观认为表示研究应与"启发式搜索"联系起来考虑。启发式搜索是表示理论的重要组成部分,其理由是既然表示是对自然世界不完善的描述,则保真推理就会将这种不完善带入其推出的结果中。从数学角度而言,这个推理是正确的,但它可能与自然世界的现象不符。另一个理由则可能更重要,本体论表示观认为合法推理可以给出问题的全部解答,但推荐推理则将给出合理的解答。这里,"合理"这一关键词有两种含义:其一是指相对小的解集台,其二是指在推理过程中大大减少的搜索空间。

(4) 本体论表示观认为计算效率无疑是表示的核心问题之一,这是这种表示观考虑"启发式搜索是表示研究不可分割一部分"的必然结论。本体论表示观强调启发式方法对表示的作用,这意味着有效的知识组织及领域有关的启发式知识是其提高计算效率的手段,但这个结论可能有失全面。

(5) 本体论表示观认为使用哪种语言作为表示形式并不是最重要的,它强调为了刻画自然世界的丰富性集成多种表示方法是必然的。另外,这种表示观特别指出表示不是数据结构,这是它与知识工程表示观的重要区别之一。

3. 知识工程表示观

知识工程表示观区别于前面两种表示观,主要体现在两个方面:其一是知识工程表示观将表示理解为一类数据结构及在其上的操作;其二是对知识的内容更强调与领域相关、那些只适合于这个领域的、来自领域专家经验的知识。由此说明这种表示观更强调其工程实现性,而不甚关心对其行为的科学解释。

综上所述,认识论表示观假设,表示是对自然世界的描述,表示自身不显示任何智能行为,其唯一的作用是携带知识,表示研究与启发式研究无关。本体论表示观假设,表示是对自然世界的一种近似,它规定了看待自然世界的方式,即一个约定的集合,表示只是描述了在这个世界中,观察者当前所关心的那部分,其他部分则被忽略。知识工程表示观认为,表示是对自然世界描述的计算机模型,应该满足计算机这一实体的具体限制。因此,表示可以理解为一类数据结构及在其上的一组操作。

不同的表示观对智能模拟研究的侧重不同。例如知识工程表示观强调自然世界在计算机内部某类数据结构的映象形式及对存储内容所采用的处理方法。因此,研究知识的存储结构及其有效地使用(推理和搜索)成为这种表示观研究的主要任务,这种表示观侧重于“计算机可接受”这个条件。对认识论的表示观而言,表示是一种携带知识的理论,问题求解的有效性不在其考虑之列,强调对自然现象(如常识知识)抽象、简洁的刻画。本体论的表示观则认为任何表示均是不完全的知识理论,而对其使用的有效性(计算困难程度)则是先决条件。因此,本体论的表示观强调一种聚焦的功能,“启发式”成为研究的一部分。

这些表示观是从不同角度及不同描述层次解释表示的内涵而产生的不同的结论。但是,本体论表示观不能因为其强调表示的不完善及可计算而否定其知识携带作用,它与认识论表示观的区别仅仅在于这种作用是否是唯一的。另外,由于本体论表示观承认表示与“启发式”研究之间的关系,因此与知识工程表示观紧密相关。

一般地,认识论表示观强调知识的某种存在性研究,本体论表示观则更多考虑知识的构造性研究,而知识工程表示观则以知识系统的可实现性作为重点。显然,对任何一门学科,存在性、构造性及可实现性都很重要,简单地否定某种表示观是不合适甚至是错误的。

11.3.2 知识表示方法

AI 中经常使用的知识表示方法几乎都是源于研究者对智能行为在微观与宏观不同层次的观察和分析而抽象出来的模型。根据表示方法的原理可以分为三类,如图 11.1 所示。

(1) 局部表示包括逻辑、产生式系统、语义网络、框架、脚本和过程等。

(2) 分布表示包括基因、联接机制。

(3) 直接表示包括各种图形、图像、声音及人造环境等。

图 11.1 中,局部表示是 AI 研究最充分也是正统 AI 最经常使用的表示方法,包括逻辑、产生式系统、语义网络、框架、脚本、过程等。一般地,局部表示又分为陈述表示和过程表示两种。陈述表示是对事物状态、属性和相互关系的描述;过程表示则是对事物的行为和操作、问题的求解方法和步骤的具体描述。分布表示方法是对局部表示在智能行为描述上不够充分而进行的补充,包括基因,联接机制。直接表示采用与自然世界一致的表示方法,

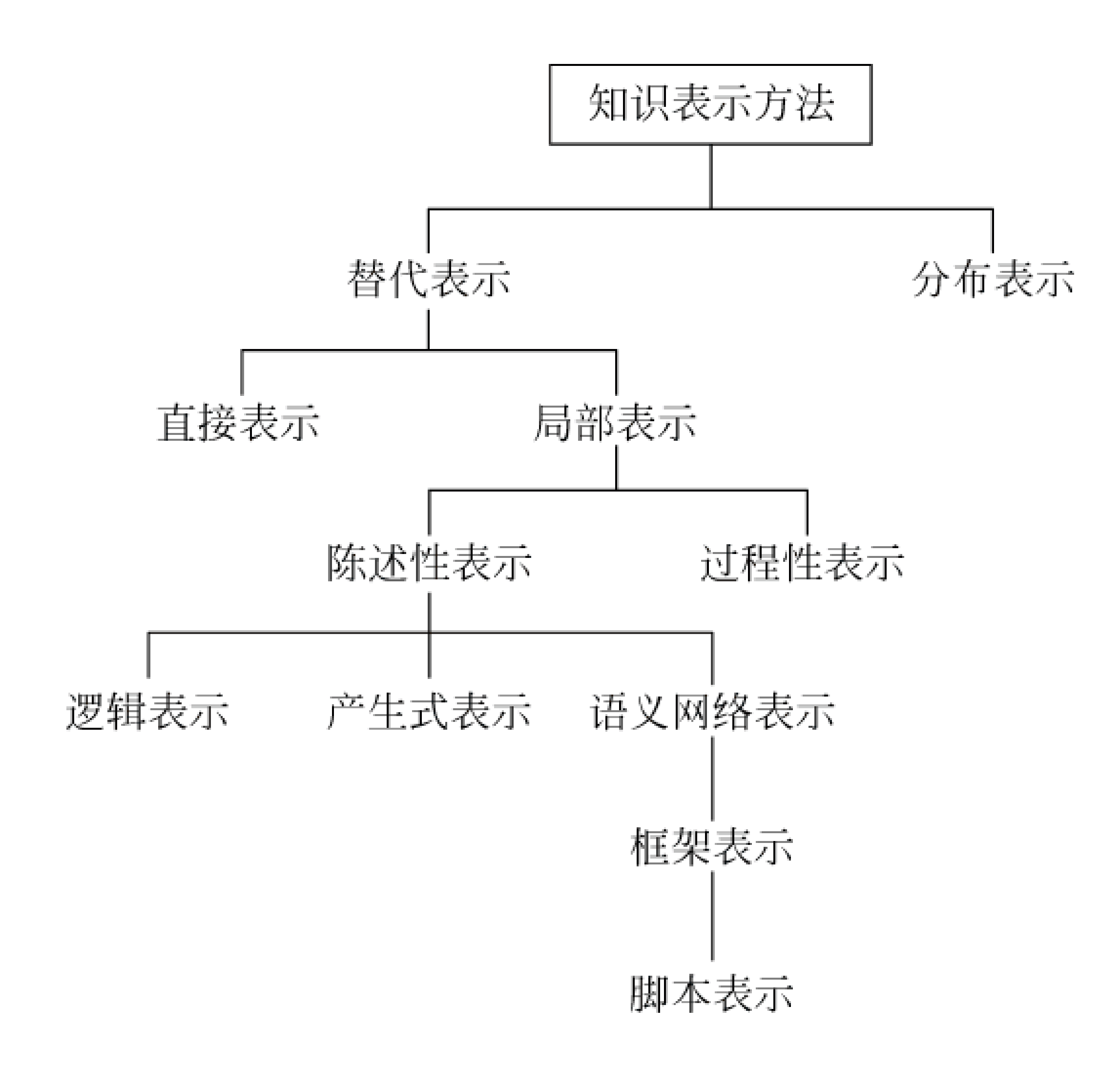


图 11.1 知识表示方法分类

早在 20 世纪 60 年代初已被提出，并引起越来越多 AI 研究者的关注。目前，这类表示方法称为直接表示或拟真表示(Direct or Analogical)，如地图、图形、图像、音乐及人造环境等。基于这类表示的系统是以对实体的拟真描述直接或间接参与推理为特点。如果考虑以计算机作为载体对知识编码，则直接表示不是一种可以完全独立于局部与分布表示的方法，主要原因是考虑到任何表示方法必须可以被计算机接受这一先决条件，因此直接表示的方法需要借助局部或分布表示的形式。对计算机而言，相对于局部和分布表示，直接表示可以视为外部表示，与其他内部表示相比较，它强调表示与被表示实体之间具有结构相似性。由于这种表示方法所存在的固有缺点与技术条件的限制，在较长的时间内没有得到长足的发展。主要原因在于：

- 计算机对直接表示的信息难以处理。直接表示的信息(如图形)具有很强的领域相关性，这暗示这种表示方法包含太多冗余信息，因此注意力集中成为必须考虑的问题。另外，大多数直接表示的信息的语义取决于其使用背景，而不是独立的。这样难以发展成为一种一般性的描述语言。
- 直接表示难以表示定量信息，换言之直接表示描述自然世界的信息范围相对受限，这使很多研究者试图设计基于直接表示的语言均以失败而告终。

下面简要介绍几种局部表示方法。

1. 产生式系统

自然界的各种知识单元之间存在着大量的因果关系，这些因果关系或者前提与结论的关系，采用产生式(或称规则)表示非常方便。实际上，谓词公式的蕴涵关系就是产生式的特例，如“天下雨，地上湿了”。

一个产生式系统通常由三部分组成，即：

(1) 一组规则，亦即产生式本身。每个规则分为左部(LHS)和右部(RHS)。一般而言，左部表示情况，即什么条件发生时此产生式被调用；右部表示动作，即此产生式被调用后所做的动作。在核实左部情况时，通常采用匹配的方法，即查看当前数据基中是否存在规则左部所示的情况。如果存在则匹配成功，否则匹配不成功。匹配成功时执行右部规定的动作，

动作一般是指对数据基中的数据进行某种处理，例如添加（增加新数据）、置换（替换旧数据）和删除（删除旧数据）等。

产生式是专家系统中使用最广泛的一种知识表示法，能够模拟人类求解问题的思维方式，便于表达专家领域的启发式知识或经验知识。

产生式规则（Production Rule）通常用于描述事物之间的一种因果关系，其基本形式是：

$$\text{IF } \langle P \rangle \text{ THEN } \langle Q \rangle$$

其中，P 是产生式的前提，用于判断该产生式是否可用的条件，也称为前件；Q 是产生式的结论或操作，用于当前提 P 满足时，应该得出的结论或执行的操作，也称为后件。例如，规则 IF 某动物吃肉 THEN 它是食肉动物（表示一种结论），IF 炉温超过上限 THEN 立即关闭风门，通知管理员（表示一种操作）。

（2）数据基（Data Base）。每个产生式系统都有一个数据基，其中存放的数据既是构成产生式的基本元素，又是产生式作用的对象。数据基常译作数据库。但这里所指的数据基和数据库管理系统中的数据库是两个不同的概念。这里，数据是广义的，可以是常量、变量、多元组、谓词、表结构和图像等，往往指一个事实或断言，可以把它看成一个知识元。

（3）解释程序。负责整个产生式系统的运行，包括规则左部和数据基的匹配，从匹配成功的规则（可能不止一个）中选出一个执行，解释执行规则右部的动作，并择机结束产生式系统的运行等。

2. 语义网络

语义网络（Semantic Network）是由 J. R. Quillian 在 1968 年在研究人类联想记忆时提出的一种心理学模型，他曾提出记忆是由概念间的联系实现的概念，把语义网络作为人类联想记忆的一个显式心理学模型。随后，J. R. Quillian 又把它用作一种知识表示方法；1972 年，西蒙在其自然语言理解系统中也采用了语义网络表示法；1975 年 G. G. Hendrix 对全称量词的表示提出了语义网络分区技术。

语义网络是对对象及其属性分类知识编码的图形结构。语义网络是一种由结点及结点间带标记的连接弧组成的有向图，其中结点表示事物、对象、状态和概念等，有两类；连接弧表示结点间的关系，有三类，可用标记说明具体的语义关系。

两类结点分别是：

- 由关系常量标识的结点，对应分类类别或属性；
- 由对象常量标识的结点，对应领域对象。

三类连接结点的弧分别是：

- 子集弧（又称 is-isa 连接）
- 集合从属关系弧（又称实例连接）
- 函数弧

语义网络是一种表达能力强而且灵活的知识表示方法，丰富的语义关系不但使语义网络能够方便地表示事物的属性和状态，还能恰当地表示事物之间的关系。目前，语义网络已被广泛应用于专家系统、自然语言理解等人工智能领域中。

一个语义网络示例如图 11.2 所示。

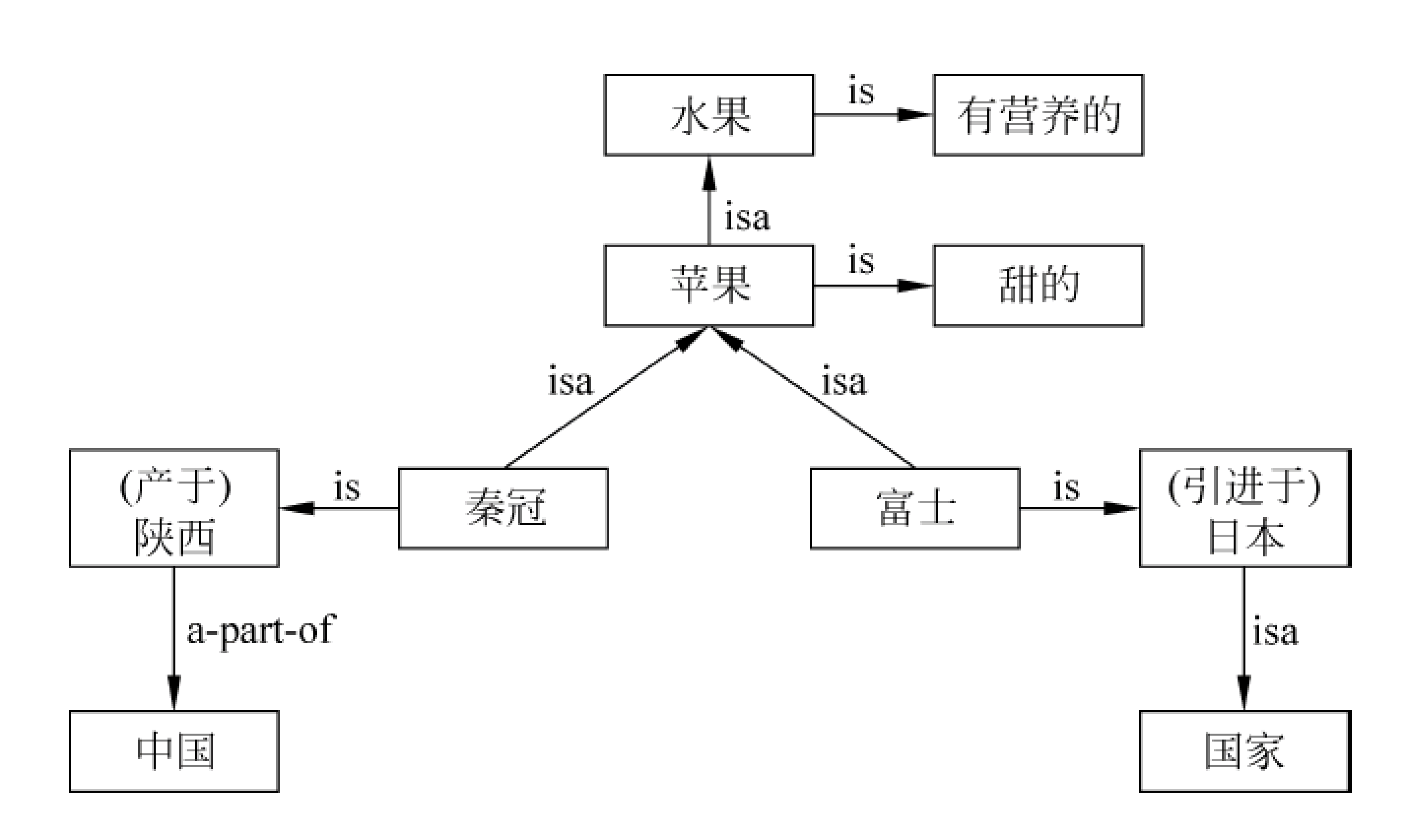


图 11.2　语义网络的示例

3. 概念图

概念结构(Concept Structure)是由美国的 John F. Sowa 提出的基于语言学、心理学和哲学为一体的一种最新的知识表示方法,不但能够表示传统知识表示方法所表示的知识,而且具有表达能力强、直观、可靠性好、易于实现、接近自然语言等特点。因此,自提出后,就在美国得到了很高的评价。

概念图的形式化定义为 CG=(Concept,Relation,F),其中:

- Concept=$\{c_1,c_2,\cdots,c_m\}$是概念结点(Concept Node)的集合;
- Relation=$\{r_1,r_2,\cdots,m_l\}$是关系结点(Relation Node)的集合;
- F(Concept×Relation)U(Relation×Concept)是弧的集合。

概念图以图形表示是一种有向连通图,包括概念结点和概念关系结点两种。弧的方向代表概念结点和概念关系结点之间的联系。概念结点表示问题域中的一个具体的或抽象的实体,概念关系结点表示概念结点之间的联系。

概念图中,概念结点用方框表示,概念关系结点用圆圈表示,有向弧标出了概念关系结点所邻接的概念结点,例如 A girl, Sue, is eating pie fast 对应的概念图如图 11.3 所示。

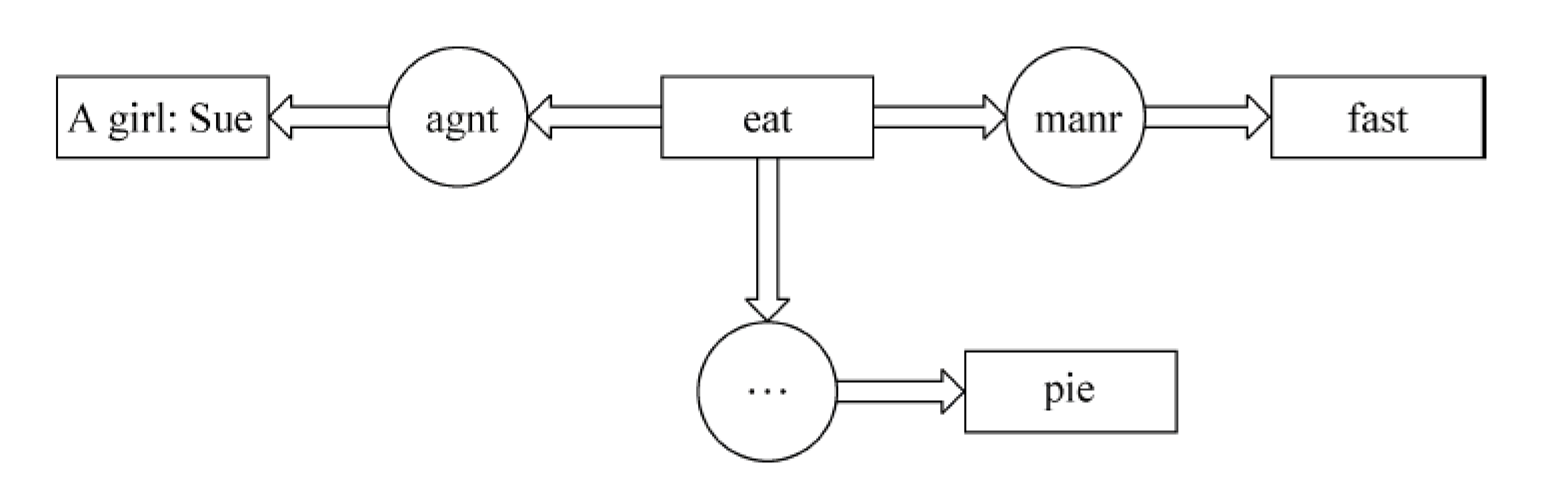


图 11.3　概念图的示例

此外,为了便于终端输出,还可用方括号表示概念结点,圆括号表示关系结点。概念图的这种表示形式也称为线性形式(Linear Form)。上面的例子采用线性形式可表示为:

```
[eat] - (agnt) - [girl: Sue]
(object) - [pie]
(manr) - [fast]
```

一个概念结点可以有两个域,其中一个称为类标号域(Concept Label),如上例中的

girl，类标号域表示一般的、不确定的概念；另一个为所指域（Referent），如上例中的 Sue。所指域表示具体的概念，可以是一个特定的值或值的集合。实际上 referent 是 concept type 的具体值，如上例的 girl 泛指女孩，而 Sue 表示一个特定的名叫 Sue 的女孩。由此可以看出引入所指域后，概念被限定为一个确定的值。另外，概念之间具有类层次关系（Type Hierarchy Relation），如 person 是 animal 的子类（Subtype），animal 是 person 的超类（Supertype）等。

例如已知概念结点的类标号集为{animal，wild-animal，pet，tiger，carnivore，feline，wild-feline，lion，jaguar}，其类层次关系如图 11.4 所示。

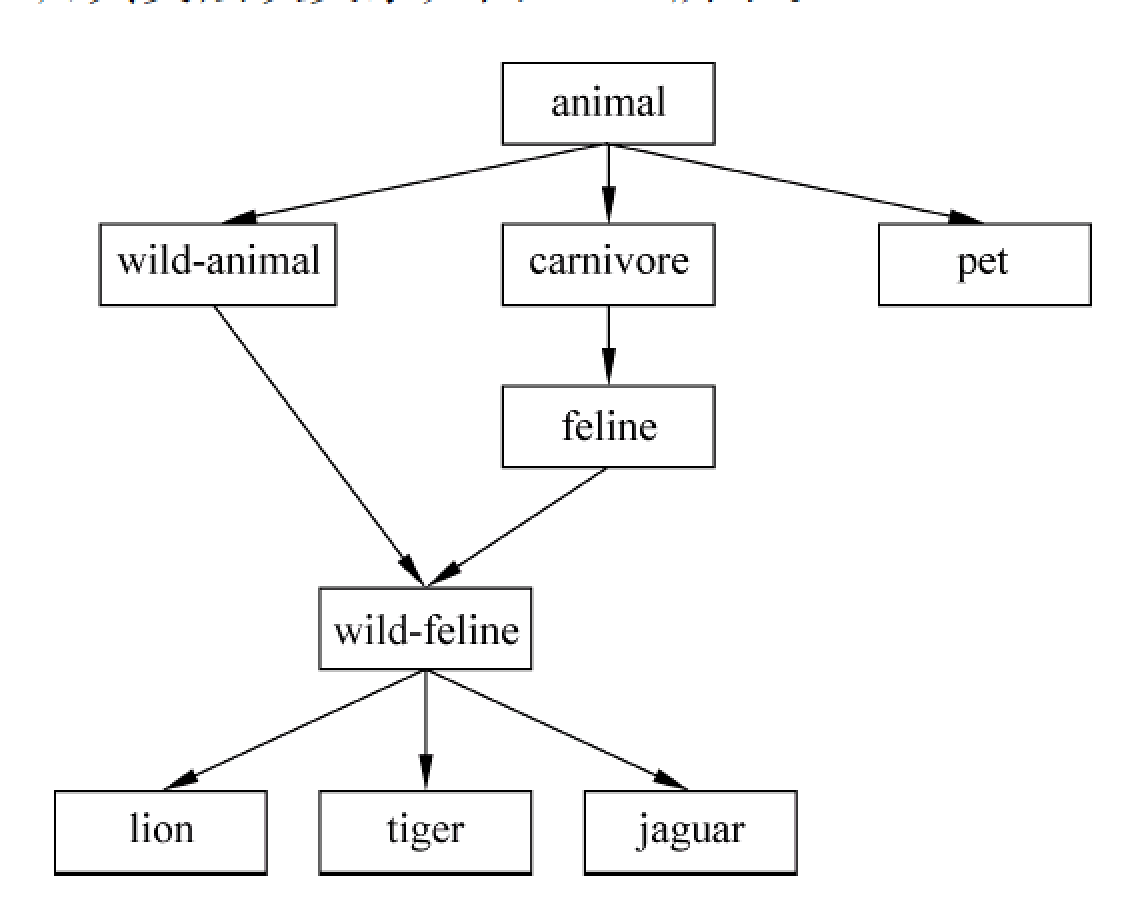


图 11.4　概念结点的类层次关系

4. 框架

框架通常由描述事物各方面的槽组成，每个槽可以有若干个侧面，而每个侧面又可以有若干个值。框架的一般结构是：

<框架名>
<槽 1><侧面 11><值 111>…
　　　<侧面 12><值 121>…
　　　　…
<槽 2><侧面 21><值 211>…
…
<槽 n><侧面 n1><值 n11>…
　　　　…
　　　<侧面 nm><值 nm1>…

较简单的情景是用框架表示诸如人和房子等事物。例如一个人可以用职业、身高和体重等描述，因而可以用这些项组成框架的槽。当描述一个具体的人时，再用这些项的具体值填入到相应的槽中。如表 11.2 所示为描述 John 的框架实例。

表 11.2　框架的简单实例

JOHN	Isa PERSON
Profession	PROGRAMMER
Height	1.8m
Weight	79kg

框架是一种通用的知识表示方法，对于如何运用框架还没有一种统一的形式，常常取决于各种问题的不同需要。

框架是一种复杂结构的语义网络。因此语义网络推理中的匹配和特性继承在框架中也可以使用。此外,由于框架用于描述具有固定格式的事物、动作和事件,因此可以在新的情况下,推论出未被观察到的事实。

框架包含所描述的情况或物体的多方面信息,包含物体必须具有的属性。在填充框架的各个槽时,要用到这些属性。框架描述它们所代表的概念的典型实例。

下面,给出一个定位零件源框架,如图 11.5 所示。

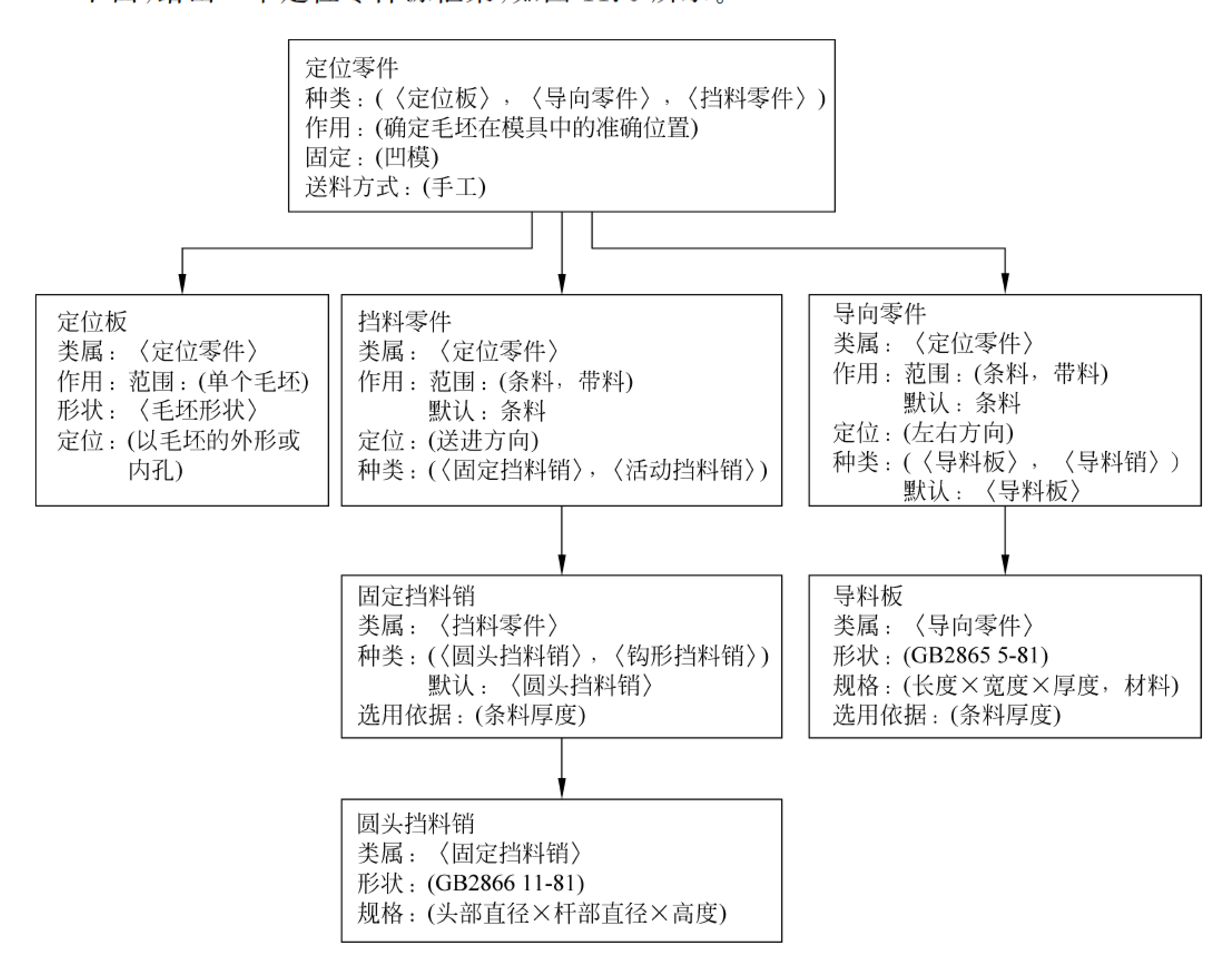


图 11.5 定位零件源框架

用一个框架具体体现一个特定情况的过程,经常不是很顺利。当这一过程碰到障碍时,经常不必放弃原来的努力去从头开始,有很多办法可供参考。

(1) 选择和当前情况相对应的当前框架片断,并把这一框架片断和候补框架进行匹配,选择最佳匹配。

(2) 尽管当前的框架和需求描述的情况之间存在不相匹配的地方,但仍然可以继续应用这一框架。

(3) 查询框架之间专门保存的链,以提出应朝哪个方向进行试探的建议。

(4) 沿着框架系统排列的层次结构向上移动(即从狗框架→哺乳动物框架→动物框架),直到找到一个足够通用,并且不与已有事实矛盾的框架。

11.4 知识可视化

Eppler M. J. 和 Burkard R. A. 认为知识可视化(Knowledge Visualization)是在科学计算可视化、数据可视化和信息可视化基础上发展起来的新兴研究领域,应用视觉表征手段,促进群体知识的传播和创新。一般而言,知识可视化领域研究的是视觉表征在提高两个或两个以上人之间的知识传播和创新中的作用。因此,知识可视化指的是所有可以用来建构和传达复杂知识的图解手段。除了传达事实信息之外,知识可视化的目标在于传输见解(insights)、经验(experiences)、态度(attitudes)、价值观(values)、期望(expectations)、观点(perspectives)、意见(opinions)和预测(predictions)等,并以这种方式帮助他人正确地重构、记忆和应用这些知识。目前常用的知识可视化工具包括如下几种。

1. 概念图(Concept Map)

概念图是康乃尔大学的诺瓦克(J. D. Novak)博士根据奥苏贝尔(David P. Ausubel)的有意义学习理论提出的一种教学技术。根据诺瓦克博士的定义,概念图是组织和表征知识的工具。它通常将某一主题的有关概念置于圆圈或方框之中,然后用连线将相关的概念和命题连接,连线上标明两个概念之间的意义关系。概念图使用结点代表概念、连线表示概念间关系,由包含一个概念的结点及连接组成。连接被贴上标签并用箭头符号指示方向,被贴上标签的连接解释结点之间的关系,箭头描绘关系的方向,"概念—连接词—概念"这样一个三元组形成了一个命题。另外,概念图是具有层次结构的,最高级的概念处在顶端。人们可以用适合的关联词说明不同层次的概念之间的关系,并确定不同分支之间的横向联系。

概念图这一知识可视化方法最大的优点在于对知识的体系结构(概念及其概念之间的关系)一目了然地表达出来,还突出表现了知识体系的层次结构。

此外,概念图还是很好的结构化知识评估工具。为了方便使用计算机辅助创建和评估概念图,美国评估、标准和学生测试中心(Center for Research on Evaluation, Standards, and Student Testing,CRESST)对概念图的概念和连接词进行了预定义,提出了知识地图(Knowledge Maps)。

随着计算机技术的发展,人们开发了一些用于绘制概念图的计算机软件工具,如Inspiration。

2. 思维导图(Mind Map)

思维导图最初是20世纪60年代英国人托尼·巴赞(Tony Buzan)创造的一种笔记方法。托尼·巴赞认为传统的草拟和笔记方法有埋没关键词、不易记忆、浪费时间和不能有效地刺激大脑四大不利之处,而简洁、效率和积极的个人参与对成功的笔记有至关重要的作用。在草拟和笔记的办法成效越来越小的情况下,需要一种可以不断增多回报的办法,这种办法就是思维导图。尽管思维导图的初衷只是为了改进笔记方法,其作用和威力还是在日后的研究和应用中不断显现出来,被广泛应用于个人、家庭、教育和企业。

托尼·巴赞认为思维导图是对发散性思维的表达,因此也是人类思维的自然功能。他认为思维导图是一种非常有用的图形技术,是打开大脑潜能的万能钥匙,可以应用于生活的各个方面,其改进后的学习能力和清晰的思维方式会改善人的行为表现。

随着计算机技术的发展，人们开发了一些用于绘制思维导图的计算机软件工具，如FreeMind、MindManager。

3. 认知地图(Cognitive Maps)

认知地图也被称为因果图(Causal Maps)，是由 Ackerman & Eden 提出的，它将“想法”(ideas)作为结点，并将其相互连接起来。想法不同于概念(concepts)，它们大多是句子或段落。认知地图是以个体建构理论(Personal Construct Theory)为基础提出的，其中的“想法”都是通过带箭头的连接线连起来，但连接上没有连接词，连接线的隐含意思是“因果关系”或“导致”，且没有层次的限制。认知地图用来帮助人们规划工作，促进小组的决策。

4. 语义网络(Semantic Networks)

在心理学中，语义网络被定义为词语或概念的语义相似性或相关程度。然而，Fisher 将其定义为结点和连接组成的网络，有连接词但不严格限制在层次结构上。这样，语义网络更像概念图，而不像主流心理学和计算机科学中定义的那样。与概念图一样，语义网络以概念和有意义的、不受限的连接词为基础，形成基本的实例或命题。Fisher 认为语义网络可以被看成多维的，而非二维的。语义网络可以非常大，包含成百上千的相互关联的概念。由于它非常大，使用者在某一时刻只能看到其中的一个部分，也就是与中心概念直接关联的概念。

5. 思维地图(Thinking Maps)

思维地图是由 David Hyerle 博士 1988 年开发以帮助学习的语言。在这种语言中，教师和学生可以使用括号图(Brace Map)、桥接图(Bridge Map)、起泡图(Bubble Map)、圆圈图(Circle Map)、双起泡图(Double Bubble Map)、流程图(Flow Map)、复流程图(Multi-Flow Map)和树状图(Tree Map)共八种图，用以帮助阅读理解、写作过程问题解决、思维技巧提高。思维地图的软件工具目前也可获得。

学生使用括号图分析物理对象，在左侧的线上是整个对象的名称或图像，在右侧的第一个括号的线上是对象的主要组成部分；桥接图为学生提供了类比过程的工具；起泡图用来描述形容词(和形容短语)的使用；圆圈图用来进行头脑风暴以及通过提供情景信息呈现一个主题的先前知识；双起泡图是比较和对比的工具；流程图用来对信息进行排序；复流程图用于显示和分析因果关系；中间矩形表示的是事件，左边是时间发生的原因，右边是事件产生的影响；树形图用于对事物和观点进行归类。

知识可视化以图形设计、认知科学等为基础，与视觉表征有着密切关联。视觉表征是知识可视化构成的关键因素。如概念图是基于有意义学习理论提出的图形化知识表征；知识语义图以图形的方式揭示概念及概念之间的关系，形成层次结构；因果图是以个体建构理论为基础而提出的图形化知识表征技术。知识可视化是通过视觉表征形式促进知识的传播与创新。无论是知识可视化设计还是应用，视觉表征都是这个过程中的关键部分。因此，知识可视化的价值实现有赖于它的视觉表征形式。

11.4.1 主要技术

RIA(Rich Internet Application，丰富互联网应用程序)构成了客户端技术基础，在进行可视化设计、探索、发布、共享阶段起着关键的作用。在目标产品当中，需要采用 RIA 技术实现体系化的完备的客户端组件仓库，在具备优异表现力的同时，还要满足跨浏览器、体积

小、运行高效的要求。

RIA 结合了传统 Web 应用的部署灵活与桌面应用的友好体验两方面优点。RIA 通常是将执行环境设计为浏览器的一个插件(Plug-in),进而在其上面执行各种应用,目的是为了尽可能地与 Web 兼容。

富客户端集成了桌面应用的交互性和传统 Web 应用的部署灵活性的两方面优点,从而创建单一完整的用户体验。富客户端中“富”可理解为具有丰富的数据模型和丰富功能的界面,进而产生丰富多彩的用户体验。

富客户端的一个显著特点是服务和表现完全物理上分离。其技术上还没有成熟的统一标准,因而富客户端只能说是一种架构方式。

富客户端技术提供了一个多样化的输入方式和符合直觉的及时反馈方式,充分利用本地计算机的资源,减少网络流量,同时还保留了由服务器自动更新的派发功能。富客户端技术正在不断的完善中,并进一步扩展浏览器功能,使之提供更加高效和友好的用户接口。

1. 产生背景

胖客户端技术的部署更新难度大和瘦客户端技术不够友好的用户体验始终让开发者和用户无法回避,用户需要找到一种新技术以解决这些问题。

早期一些软件设计人员曾尝试过在开发的系统中同时应用 C/S 和 B/S 结构,这样可以发挥胖客户端和瘦客户端互补的优势。其具体操作是将应用系统的功能需求进一步分解,然后根据不同的模块需求选择合适的客户端实现。胖客户端实现那些要求丰富交互体验和快速响应的需求;瘦客户端浏览器实现那些信息收集与发布且广泛面对用户群的需求。两种结构的结合运用增加了系统开发的复杂性并大幅提高部署和实现的成本。系统需求如何分解才合理和瘦、胖客户端之间的状态与信息如何保证一致等等,这些问题都给应用系统的开发实现带来较大的额外难度,而且随着系统规模的日益增大,问题也越来越多,这种结合变得不可取。

另外一些软件设计人员却走了一条变革的路线,他们在已经积累的架构模式上寻找出路。他们认为完全丢弃已有的技术并全新开发一种技术是不可行的,正确的做法是对现有的客户端进行改良,为适应新的需求而相互取长补短。这些软件设计人员也分为两部分:一部分人着手改进胖客户端使其易于维护和部署具有类似 B/S 结构的特点;另一部分则选择了对 B/S 结构的交互功能进行加强,使原有瘦客户端也能提供快速的应用程序响应、丰富的用户交互体验以及复杂逻辑应用的实现等,这便是富客户端技术的早期萌芽。

从 2003 年富客户端的概念出现到今天被广泛关注,期间出现了很多富客户端技术和产品,如 Google 地图等。Web 2.0 标准产生与应用的同时也大大推动了富客户端技术的快速发展。众多的门户网站和电子商务企业也开始参与富客户端技术的开发与应用,富客户端技术的研究应用也进入了高潮期。

一个优秀的富客户端应用程序应该满足用户定制、如 Gmail 一样无须等待以及桌面化等特点。富客户端应用程序一般会提供日期选择器、定制窗口、选项卡、微调控制器和标尺等工具。富客户端应用程序可以调用客户端的硬件资源和软件资源,如调用微软 Office 组件中的 Word 和 Excel 等进行文档编辑和数据计算。

最早开发富客户端应用的是 Java Applet,能嵌入到 HTML 页面中并在客户端运行。

2007 年 Sun 公司发布的 Java FX 是准备替代 Java Applet 的一种技术平台。目前应用最广的富客户端技术应该是 AJAX(Asynchronous JavaScript And XML),可以说 Web 2.0 由于 AJAX 的出现而发展起来的。典型的 AJAX 应用当属 Google 的 Gmail 和 Google Map 了。为了更好满足开发者的需要,目前已经涌现出许许多多开源的 AJAX 框架和商用的 AJAX 框架供开发者选择使用。

2. 发展趋势

RIA 开发需要具备富客户端、服务器和开发工具三个要素。由于富客户端尚未发布任何标准,各种技术都在其各自的方向创新发展。富客户端技术很好地结合了 B/S 结构的优势,解决了客户端程序的安装和维护问题,也解决了传统的胖客户端对跨平台支持的局限。Java Web Start 是美国 Sun 公司最早提出的实用富客户端技术,Google 公司研究较多也较为成熟的 AJAX 技术也应运而生,此外还有改进胖客户端类型的,如 Microsoft 的 Smart Client、Eclipse 的 RCP(Rich Client Platform)技术和采用加强瘦客户端应用的,如 Mozilla 开源的 XUL 技术、Adobe 的 Flex 技术等。AJAX 与其他富客户端技术的比较,如表 11.3 所示。

表 11.3 富客户端技术常见特性比较

特性 \ 技术名称	AJAX	Java Web Start	Flex	Smart Client	Eclipse RCP	Mozilla XUL
快速的程序响应能力	较好	较好	较差	好	较好	一般
良好的跨平台能力	好	好	好	差	好	好
方便部署与维护	好	较好	好	一般	较差	好
利用本地资源能力	一般	较好	一般	好	好	较差
终端设备支持能力	较差	较差	一般	较好	较差	较差
友好易用的用户体验	一般	一般	好	较好	较好	较好
强大的开放工具支持	较差	较差	较好	好	好	较差

Java Web Start(简称 JWS)允许用户在完全脱离 Web 连接的情况下运行和管理客户端应用程序,解决了服务器独立分发和更新客户端的部署方案。JWS 是安装 JRE 时作为 JRE 的一部分自动被安装的。Java Web Start 应用程序可以选择三种方式进行启动,即在浏览器中单击链接启动、JWS 内置的应用程序管理器启动和 JWS 内置的应用程序管理器提供的历史快速访问。无论选择哪种方式启动,其应用程序运行结果是相同的,而且每次都与 Web 服务器连接来检查特定应用程序的更新版本。

Flex 技术是在客户端运行的可执行脚本。它是原来的 Macromedia 公司(现在是 Adobe 公司)旨在增强 B/S 结构表示层效果研发的 RIA 技术,该技术提供真实而友好的用户体验。Flex 拥有强大的类库,用来创建表示层,其应用程序与传统的 Web 应用程序的主要区别在于 Flex 技术(如数据格式、字段校验、过滤、合成视频、工具提示、行为效果等处理)最适合在客户端进行。Flex 应用程序过渡流畅,提供不间断的工作流。Flex 开发模型提供的虚拟机是 Flash Player,它的应用需要运行在 Flash Player 9 或更高版本上。

智能客户端(Smart Client)是微软公司.NET 平台上的新特性。Smart Client 既具备 B/S 结构更新与升级的便捷,同时还拥有 Windows 应用程序的强大功能。智能客户端是将

胖客户端应用程序功能强大的优点与瘦客户端应用程序的部署和管理便捷的优点结合起来的产物。智能客户端应用程序能够提供内容丰富且迅速响应的友好界面和强大的客户端处理能力。智能客户端可以利用客户端的硬件和软件资源，使应用程序更加紧密地与客户计算机工作环境集成。由于使用客户端上的资源，Smart Client 可以大大减少对服务器端的硬件要求。

Eclipse RCP 是 Eclipse 核心功能之一，RCP 包括大量的控件和插件机制使得 Java 客户端的开发比以往更为便捷和简单。Eclipse 不论从显示速度上还是界面友好程度上都比其他富客户端技术更胜一筹。Eclipse 平台具有强大的可定制性，其结构本身就是一个优美的插件式结构，主要功能都能以插件的方式加入。由于 Eclipse 标准化的组件模型和普及的可扩展性，摆脱了失败 Swing 和 AWT，使富客户端程序实现起来更为简单。

XUL（XML User-interface Language）是 Mozilla 使用 XML 描述用户界面的语言。同时也是一种基于 XML 的富客户端技术。XUL 被设计用于创建可移植的用户界面，基于 XUL 的应用还可以呈现漂亮的字体、图形以及快速的界面布局且 XUL 程序界面部署和定制。XML 插入到 XUL 当中较为容易，XUL 很好地兼容了 XML 语言。此外，XUL 中的文字可以略微修改即可翻译成另一个国家的语言，很容易被本地化。XUL 核心是使用 XML 描述图形界面进而减少 GUI 程序开发的工作量。

现有的富客户端技术还包括 JavaFX、Silverlight 等等。尽管目前有许许多多的富客户端技术，但是 AJAX 采用广泛，AJAX 已经是一个无处不在的富客户端，可以在不同的浏览器和操作系统上运行。它是一个跨平台、跨操作系统和跨浏览器的富客户端技术。AJAX 使 B/S 结构中客户端的设计提升到一个新台阶，并且使客户端使用变得更加友好和灵活。

3. 工作机制、原理和应用

Google 公司的 Jesse James Garrett 在 2005 年提出了 AJAX 这一概念，AJAX 是几种浏览器技术的组合，而不是新的语言，它包括 XHTML 结构标准和 CSS 表示应用、DOM（Document Object Model）的交互和动态显示、XMLHttpRequest 异步请求服务器返回 ResponseText 或者 ResponseXML。这几项技术协同将客户端和服务器间的同步应用改变为异步通信与交互。以上几种技术的整合最终将浏览器客户端加强为富客户端应用程序。

AJAX 应用能够做到友好的用户体验是其广受欢迎的原因，同时也反映了人们迫切希望解决传统 Web 应用固有的弊端。传统的 Web 工作模型是服务器每次生成新的 HTML 页面并返回给客户端的浏览器。可是在许多网站中大部分页面的内容，如页面的页头、页尾和广告等部分显示的是相同的内容，不同的仅是一些局部内容，但传统 Web 应用每次请求服务器都会以全新且完整的页面返回。这样不管是对用户的时间、带宽与 CPU 耗用而言，还是对 Internet 服务提供商的高价带宽与空间而言无疑都是一种浪费。AJAX 技术改变了传统 Web 应用程序在处理客户端的请求时一并提交的缺陷，页面内容更新时不需要全部内容都全新装载，仅是装载需要更新的部分即可。简言之，AJAX 技术是需要什么就请求获取什么，需要多少就请求获取多少。这样，既减小了数据传输产生的网络带宽开销，也缩短了用户的等待时间，随之也降低了对资源的浪费。

与传统 Web 应用模式不同的是 AJAX 在浏览器端添加了用户响应层，如图 11.6 所示。向服务器端发送请求时不必刷新整个页面，服务器端对客户端发来的请求进行处理后，将响

应发回给客户端。客户端与服务器只进行数据层面的交换,每次不用刷新界面,也不用将数据处理的工作每次都提交给服务器完成。这样即减轻了服务器的负荷,又加快了响应速度,缩短了用户等待的时间。

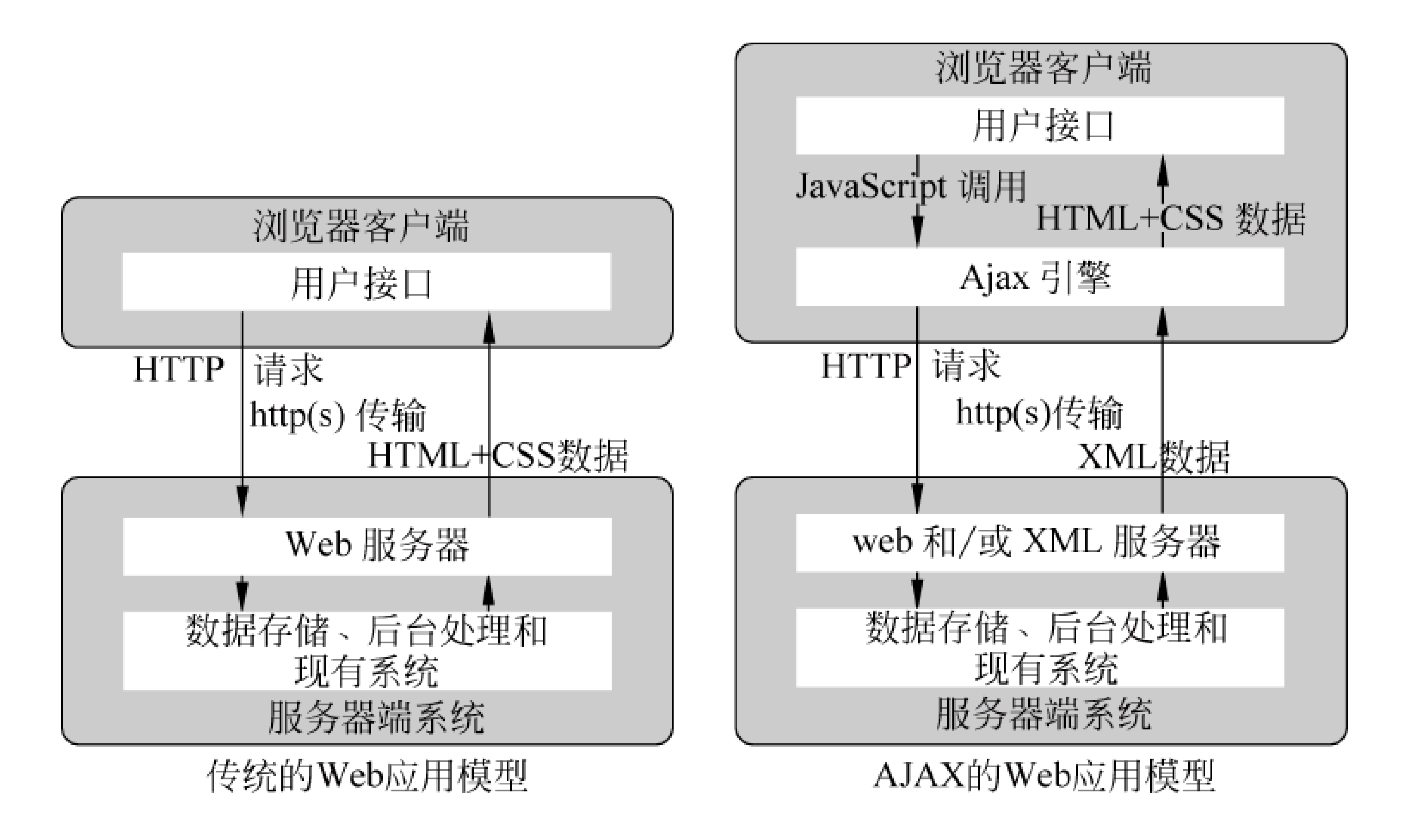


图 11.6 传统的 Web 应用模型和 AJAX 的 Web 应用模型的比较

JavaScript 设计的 AJAX 引擎,总是将客户端的请求异步发送到服务器端,然后对服务器响应返回的数据分派,再根据返回的数据是 ResponseText 还是 ReponseXML 选择相应的函数来处理。AJAX 引擎仅向服务器发送并取回必需的数据进而浏览器不需刷新整个页面就能更新页面的某一内容。浏览器也不需要等待服务器的响应,用户就可以继续在同一页面上继续操作,整个通信的请求响应过程对用户都是透明的。

AJAX 技术的核心是 JavaScript 对象 XMLHttpRequest,通过使用这个对象 JavaScript 可在不用重载页面的情况与 Web 服务器交换数据,减少了数据下载总量,缩短了客户端等待时间,减轻了服务器和带宽的负担,提高了服务器端的响应效率。严格地说,AJAX 是一种独立于 Web 服务器软件的浏览器技术。AJAX 在浏览器与 Web 服务器之间使用异步数据传输的 HTTP 请求,这样就可使网页从服务器请求少量的信息,而不是整个页面。解决了浏览器等待数据传输的问题,改善了用户体验,使用户可以快速获取 Web 服务器的数据。JavaScript、XML、HTML 和 CSS 在 AJAX 中使用的 Web 标准已被很好定义,被多数主流浏览器所支持,AJAX 应用程序具有很强的跨浏览器和跨平台性。

AJAX 引擎是在服务器与客户端之间引入中间层。客户端浏览器执行操作时便加载了 AJAX 引擎,JavaScript 调用 AJAX 引擎来产生一个 HTTP 的用户动作,负责客户端与服务器的交互。浏览器中更新的不是内容而是应用,服务器交付的也不是内容而是数据。

使用 AJAX 技术的富客户端交互方式与传统 Web 方式的不同之处在于对服务器请求方式的差异。传统 Web 应用中,哪怕仅仅是微小的交互或者需要从服务器得到简单的数据时,用户都需要等待服务器的处理。而且服务器每次都是返回一个完整的 Web 页面,这样会浪费时间和占用许多带宽。AJAX 应用避免了页面重载产生的白屏等待和数据整体一次性提交的不便,解决了传统 B/S 结构软件更新显示时必须刷新整个页面的问题。

AJAX 的核心技术是 XMLHTTPRequest,如何发送和接收 XMLHTTPRequest 的基本过程可归纳为以下六个步骤:

(1) 创建并初始化 XMLHTTPRequest 对象。

(2) 客户端发送要处理的请求。

(3) 服务器接收请求并进行处理。

(4) 服务器端返回响应数据。

(5) 客户端接收返回的响应数据。

(6) 依据响应数据修改客户端页面内容。

作为 RIA 应用开发的一类技术，AJAX 提供了丰富的用户体验。AJAX 技术有三大特点：即异步交互、基于 JavaScript 脚本和 XML 封装数据。AJAX 主要解决在现有条件下的 IE 等主流浏览器是如何在客户端提高用户体验的问题，用户体验是客户满意度的一个重要指标。从操作来看 AJAX 可以通过提供异步交互提高客户体验，其异步交互模型如图 11.7 所示(图片参考：JJ Garrett-AJAX：A New Approach to Web Applications，Feb 2005)。

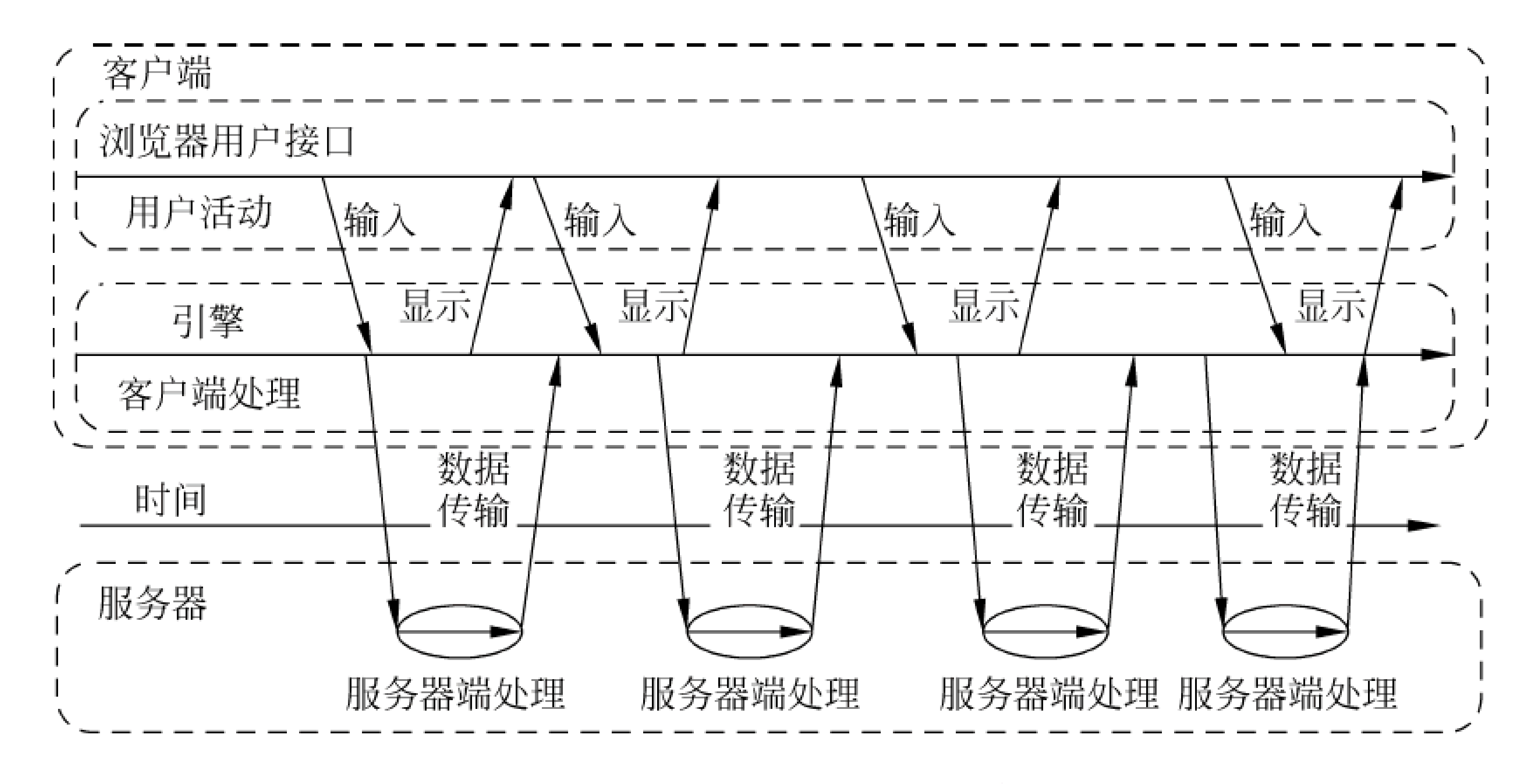


图 11.7　AJAX 的异步交互模型

ExtJS 用来开发 RIA 即富客户端的 AJAX 应用，是由 JavaScript 书写并用来创建富客户端应用界面的一个 AJAX 框架。ExtJS 可以用在 Java、PHP 等各种语言开发的各种富客户端应用中。

许多 AJAX 框架的数据交互都是基于 Json(JavaScript Object Notation)，如 JQuery、ExtJS。当下较为流行的程序语言如 VS. NET，已经将 Json 作为一个类库，可以与数据库直接进行交互，且无须配置。若是使用 Java 则需要下载一个 Json 包，然后再进行一些配置才可以使用。使用 PHP 同样也需要下载一个 Json 插件。AJAX 可以减少网络传输的内容，Json 可以将传输内容缩减至纯数据，然后利用 JQery 直接获取 Json 格式数据绑定到客户端控件上。

ExtJS 框架是基于纯 Html/CSS+JS 技术，并提供丰富的跨浏览器 UI 组件和灵活采用 JSON/XML 数据源开发，使得服务端表示层的负荷真正减轻，最终达到客户端的 MVC 三层应用。

ExtJs 支持多平台下的主流浏览器 Internet Explorer 6+ FireFox 1.5+(PC，Mac) Safari2+、Opera9+。AJAX 主流框架与 ExtJS JQuery、Prototype 和 YUI 都属于非常核心的 JS 库。Ext 是完全基于 W3C 标准的富客户端框架以实现 Web 仿真 C/S 结构的应用。

在 ExtJS 中 AJAX 返回的必须是 Json 格式，经 AJAX 处理后自动把返回的文本实例为

Json 对象,如{"age":15}形式。Json 可以减少解析 XML 带来的兼容性和性能问题。如 Google maps 就没有采用 XML 传递数据,而是采用了 Json 数据格式。

组件是数据和方法的封装且可多次利用的程序部件,一些组件经过重新封装后形成各种实用的控件。ExtJS 拥有一个结构和层次分明、功能齐全的庞大组件体系。ExtJS 组件体系可分为基本组件、工具栏组件和表单及其元素组件三类。ExtJS 开发时仅需要直接调用这些封装好的控件和组件,即可设计出如表格、窗口、工具栏、树结构和菜单等丰富多彩的应用程序界面。

ExtJS 的表格控件可以做到单选行、多选行、高亮显示选中的行、拖曳改变列宽度、按列排序、自动生成行号、支持 checkbox 全选、动态选择显示哪些列、支持本地以及远程分页、可以对单元格按照自己的想法进行渲染、添加新行、删除一行或多行和提示脏数据等。

基于 ExtJS 的开发还涉及面板(Panel)、布局(Layout)和组件(Component)等。Ext. form. Field 是所有表单输入控件的基类,对于 Ext 的表单字段来说,ComboBox、DateField、TimeField、NumberField 和 DateField 都是 Ext. form. Field 类的子类,在它上面定义了表单字段的各种基本操作和特性。在学习使用表单字段组件时,一定要重点研究 Field 这个类,掌握其主要方法、事件等,将有助于更好地使用其他的字段。

4. 优势

富客户端技术在增强良好交互体验的同时兼顾了 Web 应用零部署的优点。许多 C/S 结构软件之所以被淘汰,很大程度上要归咎于此结构下客户端程序发布与维护的复杂性。尤其是更新版本时,需要对客户端程序进行逐一下载安装和配置维护,是一件非常繁琐的事情。这也正是大量软件用户选择 B/S 结构 Web 程序的原因之一。也正是由于富客户端应用软件系统在部署成本上较为低廉,服务器和带宽开销较小,在偶尔连接服务器的无线设备上,富客户端优势尤为明显。与 AJAX 富客户端技术相比,Flex 等富客户端技术虽然有诸多优点,但需要安装插件,而且新技术的普及尚需一段时间。AJAX 在现有条件下显现出巨大的价值,目前 AJAX 已经在互联网行业得到普遍认可,友好的用户体验和 AJAX 技术的异步交互、局部刷新等特性对今后日益复杂的企业应用是不可或缺的。

AJAX 非常适用于交互频率大、频繁读取数据的大型的 Web 应用软件。基于 AJAX 富客户端技术的优势体现在以下几个方面:

- 最大程度地减少服务器资源消耗和降低网络带宽流量的开销。AJAX 的异步通信最大限度地减少冗余数据的重复提交处理,能减轻由客户端的数据申请带来的服务器应答负担,也减少了网络流量从而降低了对网络带宽的压力。
- 数据更新过程对客户端透明。AJAX 不需重新刷新整个页面即可发送异步请求到服务器,减少了用户等待时间。
- AJAX 富客户端技术具有通用性。使用 AJAX 开发的 Web 应用具有很好的通用性和适用性是由于 AJAX 基于 W3C 标准被多数浏览器所支持。
- AJAX 设计的界面友好。AJAX 的最大优点之一是可以实现实时页面更新。AJAX 技术采用异步刷新机制增强了系统的交互性,缓解了服务器和带宽资源的压力。

11.4.2 工具

Network Workbench(NWB)是一个适用于物理、生物科学和社会科学分析的集网络分析、建模和可视化为一体的综合性网络分析工具。其构架使用的是 CIShell 技术,具有分布式、松耦合、插件式等优点。新版本的 NWB 添加了若干新的算法分析文献数据,目前已经支持 ISI、Scopus 和 Google Scholar 等数据格式。CiteSpace 是用来分析和可视共引网络的 Java 应用程序,主要用于分析知识领域中的新趋势,它使用户可以将某个领域瞬时"抓拍",然后将这些抓拍的图片连接起来。如表 11.4 所示为 NWB 和 CiteSpace Ⅱ的功能比较。

表 11.4 NWB 和 CiteSpace Ⅱ 的功能比较

功　　能	NWB	CiteSpace Ⅱ
数据抽取		
ISI	√	√
Scopus	√	√
Google Scholar	√	√
Medline	√	
NSF	√	√
Citeseer	√	√
Google Scholar	√	
网络衍生		
Paper-Paper(引用)网络	√	
Author-Paper(消耗/产生)	√	
网络		
文档共引网络	√	√
作者共引网络		√
期刊共引网络		√
合著者网络	√	√
合著者机构的网络		√
合著者国籍的网络		√
同现网络	√	√
主题分类同现网络		√
引用参考文献同现	√	
(书目耦合)网络		
突变监测	√	√
其他		
类别	√	√
时区		√
时间分片		√
路径查找网络	√	√
重复结点检测	√	
合并重复结点	√	√
中介中心	√	√
K 核抽取	√	
地理地图		√

可视化工具 Vizster 是一款专门用于实现社会关系网络可视化的工具,它通过对著名的朋友关系网络 Friendster 进行可视化实现了朋友关系的在线探索。同时它能给终端用户提供一个清晰的网络结构图。借助可视化图形,可以推测出社会网络成员在不远将来的联系情况。实验结果证明,通过网络可视化得出来的结果和实际情况相差无几。同样,可视化科研合作网可使用户根据搜索到的科研合作情况了解和掌握各个国家和地区的科研重点和趋势。此外,根据网上社区的个体关系,即时通信群体之间的通信情况,博客群体间的链接结构,保险网络中的投保关系及代理关系,电子邮件之间的接收、发送次数等各个复杂网络可视化图,可以从中了解个体的生活、行为、心理和消费状况等等。

通过对保险关系网络的可视化,可以直观地分析客户的消费模式,获得客户深层次的使用信息,结合客户基本属性分析,可以掌握客户更全面的信息,用于指导保险市场营销和决策,如依据挖掘得到的具有商业决策价值的用户行为模式,制定合理的资费方式,甚至发现和分析欺诈行为。打击犯罪是可视化技术的另一重要应用,通过可视化社会关系网络,能够分析出犯罪模式和帮派活动,并通过对犯罪活动的分布和频率的分析,主动地抑制与打击犯罪活动。

作为一项非常有用的技术,可视化已经受到了越来越多的商业关注。ISS 公司的 WebTAS 是一款用来分析大型异构数据的分析软件,该系统在传统数据挖掘技术的基础上,加入了链接分析、地理信息可视化和时间轴(timeline)的表示。来自 i2 公司的分析者记事本(Analyst' Note)软件主要是关注文本数据的人物、事件、机构和银行账户之间的联系,将这些实体用图的形式展现出来。

Oculus Info 公司开发了一系列多角度分析系统 GeoTime。GeoTime 是一个可视化报告数据的软件,该软件使用了可交互的 3D 视图在时间和地理上可视和跟踪事件、个体和活动。而该公司的另一款软件 TRIST system 提供了格式化、精炼、组织和查询大规模数据的功能,该软件使用了多视图的方式帮助分析人员对搜索结果进行多角度分析,包括聚类、趋势分析和比较等功能。TRIST 的检索结果可以导入到该公司的另一款产品 SANDBOX 继续进行分析,该软件可以帮助用户排序、组织和分析大规模数据,与上一款软件的目标不同,这款软件更加注重使用计算机语言和分析功能加强用户的视觉效果,使用户进行更深层次的思考。该系统可以使用交互式的可视化分析技术提供放置、移动和分组信息的功能。

GTD 数据包含从 1970—1997 年美国所有经历过的恐怖主义行动。利用该数据,美国可视分析实验室西南中心开发了一个针对恐怖主义行动的 5W(Who, What, Where, When, and Why)的可视分析系统,旨在发现和揭示恐怖主义内在的模式和关系。该系统的功能主要包括提供 5W 的交互式可视分析;支持高层次的策略分析和单个事件的战术分析;方便研究人员的研究结果和假设的交互。

Jigsaw 是一个分析文本中蕴涵的大量实体和实体之间关系的工具。可以帮助分析人员更好地评估、分析和利用收集到的文件。其主要目标是帮助分析人员高效而准确地理解大量文本报道里包含的信息。Jigsaw 提供了多种角度可视分析文档的功能,从中识别重要的实体(例如人物、地点、组织等)和它们之间的关联。系统提供了多种视图供使用者分析,包括直方图视图、日历视图、散点图视图和表格视图等。用户还可以查看经过高亮标识的文档和文档集合。总之,Jigsaw 就像一位可视化的引导者,引导分析人员找到所需的特定文档。

VADL 是美国佐治亚理工学院花费三年时间建立的一个数字图书馆，该图书馆搜集了关于可视化方面的课件、资料和文档，可以帮助教师开展可视分析的课程以及帮助学生更好地提升他们使用和开发可视分析工具的能力。该系统使用 ResualsMaps 技术将搜索获取的信息更加友好地展示出来，使用户对搜索结果有了更加深入的了解。搜索结果按照类型放在不同区域。每个结果用一个结点表示，不同类型的结果用不同颜色的结点表示。单击结点可以使焦点移动到相应的结果上。

银行系统每天产生的大量电汇记录增加了分析电汇模式、反欺诈和反黑钱行为的难度，而且欺诈行为随着时间在不断变化，依靠传统分析电汇模式来挖掘欺诈行为已经不再有效。为此，WireVis 系统提供了基于关键字的协同分析。不同的视图展示了随着时间的变化关键字和账户的关系。该系统还提供了查询与用户提供的例子相同的模式的电汇记录。利用该系统，分析人员可以发现行为可疑的交易。

11.5 知识管理

11.5.1 概述

当智能的人类与知识共同出现在这个世界时，一种对于知识的操作就此产生，即知识的管理与共享。知识管理与共享是人类社会进步和文明传承发展的必要条件。通过知识获得、交流和共享，可使人类在知识的获取过程中，避免重复工作和相同的错误。即使在现代知识型组织中，关于如何可以达到期望结果和避免犯错误的最重要的思想和理论主要是存储在人们的大脑中，而不是在计算机或书籍中。

伴随知识经济的萌芽，知识管理备受关注。但是，就像人们对知识有不同的理解一样，以知识和人为管理对象的知识管理，目前也没有一个广泛认同的定义。

按照美国戴尔集团创始人之一卡尔·弗拉保罗的说法，“知识管理就是运用集体的智慧提高应变和创新能力”，是为企业实现显性知识和隐性知识共享提供的新途径。

美国“知识的进化”作者 Verna Allee 对知识管理的定义是“帮助人们对拥有的知识进行反思，帮助发展支持人们进行知识交流的技术和企业内部结构，并帮助人们获得知识来源，促使他们之间进行知识的交流。”

Daniel E. O'Leary 认为“知识管理是将组织可得到的各种来源的信息转化为知识，并将知识与人联系起来的过程。知识管理是对知识进行正式的管理，以便于知识的产生，获取和重新利用”。这种解释着重阐明信息、知识和人在知识管理过程中的不同角色。

Wiig 指出知识管理主要涉及四个方面：自上而下检测和推动与知识有关的活动，创造和维护知识基础设施，更新组织和转换知识资产，使用知识以提高其价值。

知识管理的目标主要包括六个方面：

（1）知识的发布，以使一个组织内的所有成员都能应用知识。

（2）确保知识在需要时是可得的。

（3）推进新知识的有效开发。

（4）支持从外部获取知识。

（5）确保知识、新知识在组织的扩散。

(6) 确保组织内部的人知道所需的知识在何处。

尽管上述定义对知识管理的理解不尽相同,但都隐含了知识管理的三个基本要素,即知识生产过程管理,知识传播过程管理和知识使用过程管理。因此,知识管理是对知识的生产、传播和使用的全程监控。它以信息技术为基础,为知识的生产、传播和使用提供开放的可管理的协同工作机制。

21 世纪是知识经济的时代,今天知识已成为企业重要的生产要素。企业需要把专门知识融入组织流程和日常工作中。知识经济时代,知识资本成为企业的核心竞争力。

早在 1959 年,管理学家彼得·德鲁克在《明日的里程碑》一书中提出了"知识工作者"的概念。1962 年,美国经济学家弗里茨·马克卢普提出了"知识产业"的概念,认为"知识产业"主要包括教育、研究与开发、传播业、信息设施业和信息服务业。

20 世纪 70 年代至 80 年代,开始出现一些超文本/群件应用系统,依赖人工智能和专家系统的知识管理系统(Knowledge Management System,KMS),以及诸如"知识获取"、"知识工程"、"以知识为基础的系统"和"基于计算机的存在论"等观点。

20 世纪 80 年代,尽管古典经济学理论忽视了知识作为资产的价值,而且多数组织缺乏管理知识的战略和方法,知识(以及以它为基础体现的专业能力)作为竞争性资产的重要性已经明确。一个美国企业社团启动了"管理知识资产"的项目;有关知识管理的论文开始在《斯隆管理评论》、《组织科学》、《哈佛商业评论》以及其他刊物上出现,关于组织学习和知识管理的第一批专著也开始出版,如彼得·圣吉的《第五项修炼》和 Sakaiya 的《知识价值的革命》,国际知识管理网络(International Knowledge Management Network)在欧洲创办。

1990 年,许多管理咨询公司开始实施企业内部的知识管理项目,而且一些著名的美国、欧洲和日本企业建立了重点知识管理项目;1994 年,IKMN 又吸收了位于美国的"知识管理论坛"和其他与知识管理相关的团体和出版物,公布了对欧洲企业开展的知识管理调查的结果。2000 年,国际商业和金融界发起倡议和批准,2000 年被确认为知识管理年。

但是,知识管理与信息技术产生密切关系是在全球互联网(World Wide Web)诞生之后,信息以前所未有的速度、广度和数量进行传播,知识的淘汰速度越来越快。信息技术在为人们提供海量数据资源的同时,也带来了数据资源处理不当和"信息超载"等问题。这种"数据爆炸,信息贫乏"的现象,使人们关注的目光再次投向知识管理。

融合了 IT 技术的知识管理经历了三个阶段:第一阶段是围绕着让正确的知识和信息在正确的时候送到正确的人这样一个目标,以 IT 技术为基础建立企业知识管理系统的阶段;第二阶段则围绕着挖掘员工隐性知识的目标,事后评价、黄页、知识地图等方法和工具的应用是该阶段的主要特征;第三阶段的标志是社会网络分析(Social Network Analysis,SNA)方法和工具的应用与推广。

1. 知识

知识,作为一个被广泛使用的词,其内涵和外延因使用者不同而异。一般可区分为狭义和广义两种概念。根据韦伯斯特(Webster)词典 1997 年的定义,知识是通过实践、研究、联系或调查获得的关于事物的事实和状态的认识,是对科学、艺术或技术的理解,是人类获得的关于真理和原理的认识的总和。总之,知识是人类积累的关于自然和社会的认识和经验的总和。这就是广义的知识概念。经济合作与发展组织(Organization of Economic Cooperation and Development,OECD),为了有利于经济分析,将广义的知识按内容分为如

下四种：

（1）关于“知道是什么”的知识，记载事实的数据。

（2）关于“知道为什么”的知识，记载自然和社会的原理与规律方面的理论。

（3）关于“知道怎样做”的知识，指某类工作的实际技巧和经验。

（4）关于“知道是谁”的知识，指谁知道是什么，谁知道为什么和谁知道怎么做的信息。

从知识共享的角度，可将知识分为两类：显性知识和隐性知识。

所谓显性知识，又称“可编码知识”（Encodified knowledge）或言传性知识，指能以一种系统规范化的方式表示要传达的知识，这类知识通常是以编码化和成文的方式进行传递和保存。显性知识的特点是可表达性，可确知性，有物质载体。在 DECD 对知识的四类划分中，关于 Know-what 和 Know-why 的知识基本属于显性知识。

所谓隐性知识，又称“隐含经验类知识”（Tacit knowledge）或意会性知识，往往是个人或团体组织经长期积累而拥有的知识，指具有个性化、难以形式化、难以与他人共享的知识，来自个人的经验、直觉和洞察力，隐性知识所对应的是 DECD 分类中关于 Know-how 和 Know-who 的知识。

显性知识和隐性知识的划分突破了过去人们对知识的认识，将未经系统化处理的经验类知识给予了重视和承认。知识管理中的一个重要观点，就是富有个性化的隐性知识更具有创新性，能创造价值，应充分挖掘和利用隐性知识。

2. 知识管理

对知识管理的含义，至今还没有一个统一的说法。比较有代表性的如下：

卡尔·费拉保罗认为“知识管理就是运用集体的智慧提高应变能力和创新能力，是为企业实现显性知识和隐性知识共享提供的新途径”。马斯（Masie）认为，知识管理是一个系统地发现、选择、组织、过滤和表述信息的过程，目的是改善雇员对待特定问题的理解。UanLeI L. ULeary 认为“知识管理是将组织可得到各种来源的信息转化为知识，将知识与人联系起来的过程。知识管理是对知识进行正式的管理，以便于知识的产生、获取和重新利用”。

上海师范大学的黎加厚教授从社会和教育信息化发展的角度提出知识管理是研究人类获取、传播、共享、利用和创新知识的活动规律，管理有关知识的各种连续过程，以促进经济和社会发展的理论与实践。各种不同的见解都反映出知识管理具有以下特征：

- 强调信息向知识价值增值过程。
- 既重视知识增值的技术实现手段，又关注人的信息习惯与素养。
- 承认知识管理不同于信息管理，也不同于人力资源管理，三者之间又互相联系。
- 知识管理必须建立在信息技术基础上。

3. 知识管理的目的

知识管理能够整合组织的知识资源，将个人知识提升为组织知识。按照组织的实际需求，对组织知识进行分类和有序化处理，且根据知识之间的相互关系进行组织、存取，便于知识积累和共享，解决知识“找不到”的问题。以组织知识库为基础，新组织成员能很快地熟悉前人的工作环境，学习其他成员的经验，进而减少因成员流动带来的知识流失，增加组织知识储备，方便组织的后继者轻松获取前人积累的知识。

知识管理在实现内部资源有序化的基础上，能够提高对外反应速度，这种快速反应表现

在组织部门内部、部门之间以及组织与组织之间。

知识管理有助于组织从现有数据中挖掘有用知识、增强组织智能。信息化的推进让组织积累了大量的数据,因此建立充分利用这些数据的意识,从零乱的数据中挖掘有用的知识,就意味着组织开始向知识管理迈进。

知识管理有助于发现和培育新的想法和思维。对于许多组织而言,一个很重要的问题就是如何使组织成员一起跨越时间和地理的界限,献计献策,交流思想,共同创造新的思维。组织通过知识管理把成员聚集到“真正的”知识共享流程中,充分利用人们头脑中的创造性思维,产生新技术和新服务的创新思维。

11.5.2 模型和技术

1. 知识的动态转化模型——SECI 模型

显性知识与隐性知识不是静止不变的,在知识管理中可以相互转化,是一个动态循环的过程,意会性知识与言传性知识之间的转化过程可用 SECI 模型表示——S 指的是社会化(Socialization),即通过共享经验产生新的意会性知识的过程;E 指的是外化(Externalization),即把意会性知识表达出来成为言传性知识的过程;C 指的是组合(Combination),即言传性知识组合形成更复杂更系统的言传性知识体系的过程;I 指的是内化(Internalization),即把言传性知识转变为意会性知识,成为个人与团体的实际能力的过程,如图 11.8 所示。



图 11.8 SECI 模型示意图

SECI 模型描述了意会性知识与言传性知识的转化与创造的一个螺旋模型,图 11.8 中间的螺旋线表示在 SECI 过程中知识的增长过程。

知识管理的核心是对知识的挖掘、组织管理与利用;通过知识管理技术,促进隐性知识向显性知识转化,挖掘存在于个人或企业组织中的富有个性化、创造性、有价值的隐性知识,促进知识的交流,实现知识的最大化共享。知识管理是综合利用管理方法与信息技术工具对知识与知识过程进行管理,以促进知识的创新、存储、共享与有效利用,提高创造与利用知识的能力,提高企业或个人自身的价值与竞争力。

对知识的管理应包含知识内容(知识资源)的管理与知识过程的管理两个方面,而两者相互促进,密切联系在一起。知识过程分为:知识组织/存储、知识创造、知识传输/共享以及知识运用四个基本环节。知识管理以知识内容管理为基础,通过知识过程管理使知识内容资源不断更新与扩充。

2. 知识管理策略

根据知识的不同特点,知识管理策略可分为显性策略和隐性策略,分别对应于显性知识和隐性知识的管理。

显性策略与隐性策略的主要区别在于：显性知识的管理策略侧重于知识的再利用；隐性知识的管理策略侧重于知识的挖掘以及人与人之间的交流。在知识管理过程中显性策略与隐性策略常结合在一起。

用显性策略对显性知识的管理策略为：对于可以整理和储存的显性知识，经过描述、加工、整理，并与相关的其他知识建立联系，对知识归纳分类后，纳入知识库中，以进行交流和共享，通过对知识的再利用，使现有知识为企业集体或个人创造价值，并创造出新的知识，为整个知识管理循环提供动力。

用隐性策略管理隐性知识的策略为：让知识需要者与知识拥有者通过各种方式与形式进行直接或间接接触，通过交流与共享，挖掘隐性知识，将发现的具有创造性的、有价值的信息进行编码，促进隐性知识向显性知识转化，把有价值的隐性知识外化后纳入知识库，通过对隐性知识进行管理，促进新知识的创造。知识管理实施的结果构成循环动态流动的知识库，知识管理需要一个用来获得知识、信息和经验的外部知识库，以及一套对知识进行过滤、组织管理的方法，以及进行广泛传播管理的工具。

3. 知识管理技术

对于知识管理而言，技术虽然只是知识管理实施过程的一环，但却是使知识管理得以具体实现的主要工具。用一句通俗的话说，就是“有了技术不是万能的，但没有技术却是万万不能的”。从技术角度来看，知识管理就是要通过科技的辅助将正确的知识在正确的时间传给正确的人。知识管理的各种功能及服务最终都还得依靠知识管理技术来实现，如搜索引擎服务就离不开搜索引擎技术、知识生产服务也需要内容管理技术的支撑。可以说，没有强大的知识管理技术支持，企业将很难有效实施知识管理，它是构建知识管理系统的基础，也是实现知识管理的强大推动力。

从广义的角度看，知识管理技术并不局限于基于计算机的技术，其他如编辑出版技术、发行技术等均是企业进行内部知识传播的传统技术，并持续发挥着重要作用。但现代信息技术才是知识管理产生的真正催化剂，也是知识管理得以有效实现的基本前提。因此这里仅将知识管理技术定位为基于计算机的现代信息技术。

说到知识管理技术，就不可避免地要将其与传统的数据管理及信息管理技术进行比较。

数据管理技术是指那些能够协助人们生成、检索和分析数据的技术，以数据为管理对象，通常处理事实、图形等原始资料，典型的数据管理技术包括数据仓库、数据搜索引擎和数据建模工具等。而信息管理技术是指能协助人们更好地处理信息的技术，以信息为处理对象，如自动化信息检索与查询系统、决策支持系统、经营信息系统、文档管理技术等。无论数据管理技术还是信息管理技术，其处理对象大都是显性的，对隐性的知识基本无能为力，无法把握知识的丰富性和知识背景的复杂性，也难以有效支撑知识管理所强调的知识分享和协作功能。

虽然，知识管理技术和数据管理及信息管理技术有着区别，但需要强调的是，知识管理技术并不排斥数据管理及信息管理技术，反而是建立在数据管理及信息管理技术的基础之上，针对知识特性而开发的、能够协助知识工作者进行知识生产、分享、应用以及创新的技术，是现代信息技术在知识经济时代的新发展。传统的数据管理及信息管理技术仍然会在知识管理中得到进一步的广泛应用，成为整个知识管理技术体系中的重要组成部分。

这样,就可以给出知识管理技术的基本定义。知识管理技术就是指能够协助人们生产、分享、应用以及创新知识的基于计算机的现代信息技术。从知识管理技术的定义来看,知识管理技术并不是一项技术,而是一个技术体系,包括的技术内容异常繁多,覆盖了知识生产、分享、应用以及创新的各个环节。它同时又是多种信息技术的集成,这些技术结合起来形成了整体的知识管理系统,为企业提供知识管理服务。

知识管理实现有两个基本目标:一是已有知识的共享与重用;二是创造新知识。这是跨越某一时段的实践过程,与人、商业应用和信息技术密不可分。

知识管理包括以下方面:

- 知识聚集——包括已有知识的集成和新知识的获取。知识聚集又称为知识生产。
- 知识组织和存储——给获得的知识赋予一种结构并合理地存储,以便有效管理和使用。
- 知识演变——由于知识聚集过程中发生的偏差,以及知识的时变性,需要更正、更新、删除旧知识,增加新知识。
- 知识传播——使需要知识的任何个人和组织可以在任何时间、任何地点获得知识。
- 知识使用——在知识处理系统之间和相关人员之间实现知识共享和重用。

知识管理离不开知识管理系统的支撑。知识管理系统不是一个单纯的知识发布系统,而是一个交互式的开放的协同工作环境。知识管理涉及许多核心技术,主要包括:

(1) 信息技术

与互联网、电话网和电视网等相关的信息技术主要用于解决知识的快速、高效、实时和准确传输,是知识传播的基础设施。知识传播是集语言、文字、图像和声音于一体的多媒体传输,信息技术的快速发展为这种多媒体的快速、实时和准确传输创造了条件。

(2) Web 技术

目前,Web 技术及其多层体系结构是知识管理系统普遍采用的技术。Web 技术主要涉及到两类标准:一个是应用层协议,如超文本传送协议(HTTP);另一个是表示层句法,包括超文本标记(HTML)和扩展标记语言(XML)。此外,动态页面技术可以使得用户能够访问存储在 Web 中的实时数据。

(3) 知识存储技术

知识管理需要管理的知识量非常大,这些知识需要有效地组织和存储。为了使知识能够更好地共享和重用,知识和知识处理系统应该分离,知识应该独立于知识处理系统。而且,知识通常是比较稳定的,更新的速度较慢。因此,知识库的知识时限达到几年甚至几十年。并且,知识库面对的主要问题不是查询,而是知识的分析和处理。所以,与事务数据库相比,数据仓库更适合用来存储知识。

另外,越来越多的知识以 HTML 和 XML 文件的形式存储在 Web 服务器中,这给知识存储提供了新途径。

(4) 知识获取技术

Feigenbaum 曾说知识获取是人工智能中最重要的核心问题,是人工智能研究的关键。知识获取是指从纷繁的信息中发现、提取和挖掘知识。知识获取可以分为人工获取、机器辅助的人工获取和机器自动获取三种类型。就目前的技术发展而言,机器辅助的人工获取既能保证一定的精度,又能确保较高的效率,而知识的机器自动获取是人工智能追求的目标。

机器学习是机器获取知识的主要方法,而数据挖掘和 Web 挖掘是机器学习的重要研究方向。

(5) 知识表示与本体

知识是知识管理的管理对象之一,是人类认识客观世界的创造性成果。在人类思维活动中,认识加工和处理知识的主体,供人类学习使用。因此,知识构成人类思维活动的环境,是人类创造性工作的基石。

机器要加工和处理知识首先必须拥有知识,因此,一个关键问题是机器如何表示知识。就好像人类创造了语言文字,使知识可以保存记录下来,这些知识既是人类创造的成果,又是后人学习和再创造的基础,知识表示就是机器描述知识的语言和文字。

构建智能系统的一个新方法是聚集可重用构件。今天建造基于知识的系统通常必须从头开始建立知识库,其实可以通过聚集可重用构件完成。系统开发人员只需创建与其系统任务相关的特定知识和推理机。新的系统将和已有系统进行互操作,利用它们执行某些推理。通过这种方式,知识、问题求解和推理服务可以在系统之间共享。这种方法有利于廉价地建立更大和更好的系统。然而,时至今日,这一设想还远未实现,其中一个重要原因就是知识不能共享与重用。

Angus 等指出目前的知识管理不能重用从经历中获得的知识,因为它不以一种形式化的方式共享。知识的共享与重用是知识管理的实现目标之一。可见,知识共享与重用不仅是构建智能系统的普遍性问题,更是知识管理面临的严峻挑战。知识共享与重用需要一种形式化的共享的知识表示规范,尤其是现在的知识管理系统乃至智能系统都是分布式系统,集中式知识表示已经不能满足应用的需要,必须研究知识的分布式表示方法。知识表示的基础是本体(Ontology),它是共享知识表示的关键。

为什么说本体是共享知识表示的关键呢?原因在于:首先,本体阐述了知识的结构。给定一个领域,其本体构成了面向这一领域的知识表示系统的核心。没有知识之下的本体或领域概念化,就没有一个表示知识的词汇。其次,本体使知识共享成为可能。假设我们作了一个分析,得到某一知识领域的一个满意的概念化集合的代表项。为了建立知识表示语言,我们在项与本体中的概念和关系之间建立联系,设计用概念和关系编码知识的语法。因为有了本体,就不必重复知识分析过程,而可以与他人共享这一知识表示语言。共享本体形成了领域相关的知识表示语言的基础。基于本体构造的知识表示语言是内容丰富的(或语义丰富),它们有大量包含复杂领域内容的项,这类共享将大大增加知识重用的潜力。

建立可以共享与重用的领域本体不仅是知识表示的需要,也是实现知识共享和重用的关键,是实施知识管理的一项基础工作。

11.5.3 知识管理系统

1. 知识管理系统

目前关于知识管理系统的定义还很不统一,有部分人把知识管理实施中组织内涉及的各个方面(例如著名的知识管理专家 Peter Meso 和 Robert Smith 就认为知识管理系统应该由知识、技术基础、组织基础、人力资源、文化五部分资源组成)都称作为知识管理系统;而另一部分人则把知识管理系统定义为辅助组织实施知识管理的 IT 系统,即所指的知识管

理系统是辅助组织实施知识管理的 IT 工具。

知识管理系统不等同于知识管理技术。目前知识管理系统中应用的主要技术有网络通信技术、知识仓库技术、自然语言理解技术、神经网络技术等,这些新技术能够改善知识管理系统的性能。知识管理系统技术随着技术的发展不断涌现与更新,如过去集中于文件的建立、维护和版本控制的文件管理系统,如今与群件/协作应用结合起来。

在系统的构成方面,面对越来越多的技术可能性应当慎重考虑。软件模块的功能并非越多越好,应当经过仔细的考虑和筛选,使得系统具有三个特性:实用性、友好性和可拓展性。所谓实用性,就是要保证成员能够从使用该知识系统获益,而不是给组织提供一个中看不中用的“花瓶”。所谓友好性,就是系统应该是容易使用,容易学习的,用户对系统没有畏难情绪,是系统能够迅速推广的一个重要条件。所谓可拓展性,就是系统要能够适应组织业务的变化。知识管理系统往往是随着组织需求的增加而扩大的,需要能随着它所支持的业务一起成长。在系统设计中,往往从一个核心的需求开始,逐步扩大其规模和功能。

基于以上的考虑,知识管理系统根据其模块的组合,提供知识管理的各种功能。在实际的应用中,有的组织注重实现个体间的知识共享,因此知识管理系统需要有良好的知识整理和知识传播的能力;有的组织注重协同性知识工作,通过思想火花的碰撞产生新的知识;有些组织则把重点放在对知识的捕捉、操作与定位,侧重进行与知识相关的信息管理。另外一些组织着眼于建设、开发智力资本,提供自由的、不受限制的、简单易行的对话功能,以提高组织中知识活动的有效性。还有一些组织注重创造一种学习环境,从而使得成员能够保持对新知识的关注。

2. 知识管理与信息管理的关系

知识与信息,知识管理与信息管理是紧密相关的。信息是事物运动的状态及其变化的方式,信息有时与主体有关。而知识是客观的,与主体无关。知识是信息,是信息的提炼和一般化,然而信息不一定是知识。

信息管理是为实现组织目标、满足组织需求而对信息资源进行规划、开发、集成和利用。在知识管理与信息管理的关系方面,信息管理是知识管理的基础,知识管理是信息管理的拓展和延伸。信息管理和知识管理之间的区别体现在以下几个方面。

首先,知识管理拓展了信息管理的对象。知识管理的对象包括知识以及传统意义上知识的创造者——人,而信息管理仅仅局限于用语言、文字、图形、图像和视频等各种载体表示的信息。因为知识存在于人的头脑之中,所以只有人能够充当知识创造的主体,尽管计算机具有强大的信息处理能力,它只能是知识创造的辅助工具。因此,知识管理把人纳入了其管理的范畴。

其次,知识管理提升了信息管理的地位和作用。信息管理服务于组织的生产、经营和管理。信息管理系统是为组织整体的管理、控制和决策等服务,它的成功运行可以保证组织运作的高效率、及时性和适应性。在知识型组织中,知识的生产、传播和利用是组织的主要工作之一,是组织核心竞争力的综合体现,是组织在激烈的全球竞争环境中制胜的法宝。知识管理不仅仅是人与人之间的知识共享,还包括知识的创造和利用。创造知识的增值价值和增值服务是知识管理的目的,也是其备受青睐的原因所在。

再次,知识管理强化了对信息管理基础设施的要求。特别是在知识传播方面,要求做到任何时间,任何地点,任何人都可以获得所需要的知识,以保证知识服务的实时性和高效性。

最后，知识管理是与信息技术、人工智能、管理科学、人文科学和经济学等相关的交叉学科。知识管理研究人和知识在知识型组织中的地位和作用，研究人类生产知识的机制，以及信息技术如何被用来提供知识生产、流通和使用的支撑环境。知识管理还研究知识如何有效组织、存储和处理，以协助乃至代替人类使用和创造知识。与之相比，目前的信息管理在相当程度上还属于计算机应用技术的范畴。

11.5.4 方法和步骤

1. 实施方法

知识管理实施的方法论包括战略、实施以及保障这三个层面，具体包括：

(1) 构建支持知识管理的组织体系。知识管理具有责任分散的趋向。因此，要建立一套有效的组织体系，以支持企业的知识管理活动。在这一体系中是要有负责知识管理活动的领导人，承担制定管理计划和协调企业的各种知识管理活动；即成立专门的小组，完成与知识管理活动有关的任务。他们可以来自不同的部门，以保证各个部门的需要可以都得到协调和满足。此外，企业领导人的支持是知识管理活动成功的保证。

(2) 构建支持知识管理的技术体系。要建立一套有效的技术体系以支持企业的知识管理活动。主要是建立支撑知识管理的基础设施，如统一的信息技术平台、数据库和知识库等，并采用专业的知识获取、知识表示、知识发现软件。知识管理实践实际上是一个信息技术平台重新综合、建立的过程。一个良好的知识管理技术体系，可以提升知识管理的效率，易于知识管理在实体内的推广。

(3) 加大对知识管理的资金投入。企业知识管理活动需要资金支持，这反映了企业对知识管理的责任感。要动员企业全体人员为知识管理投资，以保证知识管理活动的正常开展。

(4) 创造知识共享的企业文化。有利于知识共享的企业文化，包括良好的职业道德、企业荣誉感和团队精神等等。发动全体员工，营造知识管理的氛围，建立学习型的组织。教授员工学习和掌握知识库的使用方法，鼓励员工善于利用知识库，特别要鼓励员工整理固化知识并将知识上传到相应的知识库，鼓励全体员工贡献自己的知识与他人分享，从而不断丰富和完善企业的知识库，形成全员知识管理的企业文化。

(5) 建立动态联盟，培养知识创新能力。知识创新是知识管理的终极目标，没有知识创新能力的公司将很快被市场淘汰。通过建立动态联盟、构建战略合作伙伴的知识网络体系，是提高企业知识创新能力的有效支持手段。

(6) 建立知识管理评估体系。不要用传统的财务方法对企业的知识管理活动效果做出评估。而是要研究和建立面向知识经济和知识管理的无形资产评估体系，如无形资产组成指标法、计算知识管理的投资回报率等。

2. 实现步骤

知识管理实现可以分为战略规划阶段、业务规划阶段、系统实施阶段和评估与改进阶段，具体步骤如图 11.9 所示。

(1) 战略规划阶段。包括知识管理目标与愿景的制定、现状评估差距分析及策略制定。知识管理的目标不能脱离实体的管理目标独立存在。它必须与实体的总体战略目标一致，

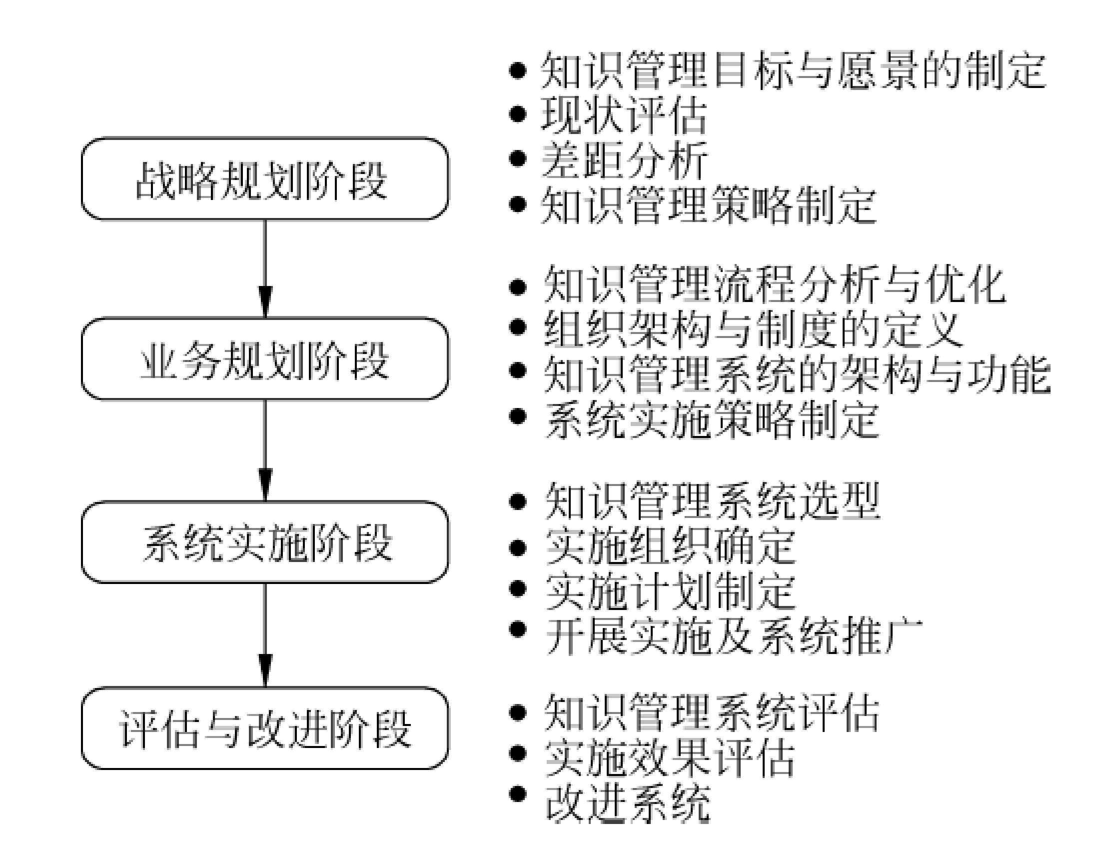


图 11.9　知识管理实施阶段的具体步骤

才能有生存和发展的基础。现状评估及差距分析应以知识管理对实体的支撑和提升为核心,从实际现状和需要,结合国内外知识管理实践,对知识管理如何促进实体的管理和运营进行资料收集、调研、讨论和分析。从而制定知识管理的策略。

(2) 业务规划阶段。包括知识管理流程分析与优化、组织架构制度的定义、知识管理系统的架构与功能、系统实施策略制定等方面。知识管理的流程应包括知识的获取、组织、重构、存储或记忆,以及分配、共享、交流和创新、应用。业务规划阶段应考虑实体的知识管理建设的短期和长期的目标。例如,知识检索系统,应能为实体提供更智能化的知识检索,而不是简单的一种搜索结果的堆积,应侧重于知识的检索、传播和利用。

(3) 系统实施阶段。包括知识管理系统选型、实施组织确定、实施计划制定、开展实施及系统推广。每个实体都有不同的业务体系,包括生产、研发、销售等,不同的业务体系的任务特性均不相同,其完成任务所需的知识亦有不同,因此需要根据不同业务体系的任务特性和知识应用特点,拟订最合适、成本最低的知识管理方法。另外,考虑到一种业务体系下有多方面的知识,如何识别关键知识,并判断关键知识的现状,在知识管理模式的指导下采取有针对性的提升行为。所以,此阶段的重点是结合企业业务模式进行知识体系梳理,并对知识梳理结果进行分析,以确定知识管理具体策略和提升行为。本阶段是知识管理从战略规划到落地实施的阶段,根据对企业试点部门的知识管理现状、需求和提升计划的分析,应该考虑引入支撑知识管理落地的知识管理 IT 系统。根据前几个阶段的规划和分析,选择适合企业现状的 IT 落地方法,如带知识管理功能的办公协同系统、知识管理系统、知识门户系统等等。

(4) 评估与改进阶段。对企业的知识管理系统的本身及实施进行评估,并根据企业实际进行改进。在当今信息化的社会中,信息的更新速度和市场环境的变化速度比以往任何时候都快。所以要不断地评估现有的知识管理体系,并加以改进,使之适应不断变化的市场环境、社会环境和实体的管理目标。

第 12 章　语义网和本体

12.1　语义网

1990 年 Tim Berners Lee 发明了万维网，旨在通过 Internet 获取各种信息。二十几年过去了，万维网飞速发展，从最初被动地发布数据，到交互式地获取所需数据，到现在实现智能检索，即根据用户需求获取信息。现有的检索工具，如 Google 已经把检索范围和搜索速度提升到前所未有的程度。然而，人们发现万维网还是无法满足日益丰富多样的需求，其局限性体现在：

(1) 信息是海量的，但缺乏对信息的描述，即缺乏元数据(Metadata)；

(2) 万维网的基石——HTML 提供的链接缺乏语义；

(3) 基于关键词检索的万维网搜索引擎的检索质量和效果不尽人意。

2000 年 12 月在 XML 2000 会议上，Tim Berners Lee 提出了下一代因特网的概念——语义网(Semantic Web)，为人们描绘了未来语义网的美好前景。语义网是当前万维网的扩展和延伸。语义网的信息具有充分、完备的语义定义，能够在人与计算机之间建立语义上的理解与合作。因此，语义网是具有语义的万维网，是能够理解语义的万维网。

12.1.1　概述

语义网研究的重点是如何把信息表示为计算机能够理解和处理的形式，即带有语义。语义网中语义是核心，即能够在人与计算机之间、计算机与计算机之间以无偏差的方式传递的信息。语义网的基本思想是对互联网上的任意资源，进行结构化的描述并引入语义，使得计算机可以理解互联网上的信息。当然，计算机不可能真正像人一样进行思考，但是通过制定标准，使用标准描述信息的含义，计算机就可以根据标准自动进行分析和推理，将网络服务集成在一起，使自动化智能服务成为可能。语义网是通过在网上提供定义好的、相互链接的数据，让网上的数据被高效、自动地发掘利用、不同的数据能更好地集成，而且能被各种不同的应用程序使用。

语义网提供一个基础架构，通过这个架构在 Internet 上不再只处理 Web 页面，数据库、Web 服务、程序、传感器、个人智能设备甚至家用电器设备都能通过网页来传递并处理数据。各种软件代理能够搜索并过滤这些数据，以一种全新的令人激动的方式把这些处理好的数据送到 Web 使用者面前。

回想一下 Internet 刚普及时的文档处理系统，检索并引用远程系统的信息还是专家们的游戏。虽然 Internet 可使用户很方便地登录到远程系统，然而这些系统往往使用不同的信息提取协议，例如通过 Telnet 登录到一个远程系统后，在获取信息之前用户需要首先了解该系统的信息提取协议，而且找到所需信息后，要先复制到用户的剪贴板，然后再复制(或者重新输入)到自己的文档中。采用上述方式对于处理那些关联性强、时效性和准确度要求

极高的文档简直就是一场噩梦。

有了 Web 技术后,则能轻松实现信息间的无缝链接,尽管很多系统的 Web 服务器可能运行在不同的机器上,但在 Web 应用程序之间传递内容仍然是很困难的。目前,在充分利用 Web 方面仍然受到很多束缚。假设用户在浏览 Internet 时偶然打开了一个会议召集通知,Web 上有召开会议的时间和地点,并且还有很多超链接地址,分别链接到本次会议召集人及其他参与人员的个人主页。当用户报名参加本次会议,开始单击注册按钮,此刻期待着电子日历能自动记录会议的日期和时间,并能链接到 Web 上的详细说明,希望数字电话能下载会场地址并计算出在会议当天到达的最佳行车路线;还希望随身携带的商务通能自动把参会人员的联系方式下载并临时保存起来,直到会议结束。用户是多么希望上述处理能够在 Web 上通过一次单击即可全部自动完成。

遗憾的是,现在还无法做到。事实上用户不得不非常辛苦地把会议详细情况逐条复制并粘贴到地址本,自己去查找会议日期和时间,不得不手工从各个会议参加者的个人主页中寻找并复制其联系信息到地址本中,手动调整其地址和电话号码格式,还不得不在手机上录入会议的位置信息。以上描述的情形还只是个人在网络上处理数据所遇到的麻烦,处理企业业务数据时的困难更是可想而知。如果用户试图连接公司内部运行的不同的数据处理系统,或者试图帮助客户从多种数据库中整合所需的信息,可能会遇到非常尴尬的情形。在库存管理系统和财务系统中存在很多重叠的数据,在整合这两个系统的数据时很容易发生主键冲突或者数据关联错误,可能不得不使用程序员编写的接口程序从库存管理系统中筛选并格式化数据,然后导入到财务管理系统。同时还发现企业的客户关系管理系统也应该和订单管理同步进行数据整合,否则将会严重影响公司的业务和生产。一次又一次,只能不停地让程序员编写各种接口程序,如果公司存在很多不同的应用系统,将需要编写大量代码提供各种数据接口,这无疑会带来高昂的程序维护开销。

使用扩展标记语言(XML)对于改善上述情况将有所帮助,如果所有的应用程序都采用 XML 格式,程序员只要学会处理 XML 数据,就不必和各种离奇古怪的数据格式打交道。这意味着可以利用一些 XML 工具,如 XSLT(一种转换语言,参见 http://www. w3. org/TR/xslt)来粘合应用程序。遗憾的是,这种技术还无助于彻底改善数据接口的效率。因为每一对应用程序之间,甚至同一对应用程序的每一种接口之间,都需要定制相应的 XML 到 XML 桥。换言之,在不同的应用程序之间提取 XML 文件时,不是简单地进行合并。为了执行针对 XML 文件的查询,还需针对其配对文件补充特定的限制条件,不是简单地把两个查询合并到一起。这与关系数据库中通用的数据元能被轻松地连接到一起的处理方式大不相同。

但是,不同的数据库由不同的 Schemas(数据视图)文件组成,而且这些 Schemas 的表达并不清晰。因此,仅靠 XML 标记很难直接和另一个数据库中的域关联。解决办法之一是把这些 Schemas 变得更明白易懂,并映射为统一的术语。XML-Schema 语言(http://www. w3. org/XML/Schema)允许很多公益组织整理出统一的 Schema 文件。一个公司甚至是一个特定的商业部门,通过开发一个统一的 XML 映射集(例如一个特定的 schema 文件),就能采用统一结构表达信息。实际中,实施起来并不容易,而且针对不同用户开发一个大型词汇表是非常棘手的事情。

不同结构的 schemas 文件,或者是基于不同商业词汇的不同用户的 schemas 文件之间

的映射，都不是 XML-schema 所能解决的问题。实际中经常需要处理异构数据的映射问题，为此需要寻求更为有效的数据表达工具。例如关系数据库中的关系演算，数据表达能力远远胜过许多旧的数据库（文件数据库），因此它成了过去处理数据映射的标准。更为有效的表达方法，例如实体关系或者对象模型，可以解决复杂的数据映射或者查询异构数据。总之，采用更有表现力的语言能提升协同工作的层次。既然以前老的数据系统采用关系模型很好地解决了数据兼容问题，所以非结构化的 Web 数据，或 XML-schema 定义，也可以通过关系模型有效解决数据模型问题。

为此，建立了一个名为资源描述框架（Resource Description Framework，RDF，参见 http://www.w3.org/RDF/）的语义网基础组件。如果两份来自不同数据源的 RDF 格式文件需要合并，只需要将其合并成一个大文件，即把文件中的关键字进行简单的连接。因为 RDF 文件格式的关键字均采用相同的通用资源定位符（URI）。如果想在合并后的 RDF 文件中增加限制条件，修改原来的查询方式，只需要直接在新的 RDF 文件中增加限制条件即可。XML 文件是由元件和属性组成的，只能告诉我们文件里面记录了什么内容，而 RDF 则由一段段数据表达式组成，每个表达式都描述了一个特定的值，这个值相对于一个数据库表的单元。原有的关系数据库运算都可兼容，如连接和视图等，并可以使用常见的工具加以执行。

这样可以顺利地解决企业级应用系统间的数据集成问题。只要把每个应用程序的数据输出转换为 RDF 格式文件，就可以针对 RDF 执行各种查询，可轻松地编写并修改查询条件，导出所需要数据。反之，这些数据也能轻松地导入到其他应用程序中。而且，这种问题和系统规模只是线性相关，就好像添加新的 Web 服务器不会影响到其他人浏览 Web 一样，新的 RDF 也能被轻松地添加到 Internet 上，而不会影响正常使用。大量需要人工编写的数据接口奇迹般地消失了，就像文档之间可以连接一样，数据也能通过 Web 连接在一起。

正如如果没有 RDF 就难以在互联网上整合数据库一样，应用程序的跨互联网整合也遇到了同样的问题。表面上来看，在 Internet 上整合应用程序是很容易的，经常是轻轻点击一下就从 Internet 下载 Java 或 Flash 程序到本地运行。但这对于电子商务应用程序是无效的，特别是在 B2B 的应用程序之间。

设想某个企业想从一个供应商那边购置一批零部件，需要先联系大型船运公司安排船运，然后从本地几个生产商中精心挑选一家在零件运到时具备最高生产能力的厂家生产。而且是希望能通过 Web 高效地解决这一问题，即由一个销售员下订单，然后启动整个供应链高效协同工作。这看上去和前面提及的数据库间整合有几分相似，然后却要复杂得多。因为牵涉到的各家企业采用的内部管理软件可能完全不同，而不仅仅是数据库的不一致。更糟的是，这些应用程序可能运行在企业内部某台特定用途的计算机上或者隐身在内部防火墙和安全防护设备的后面。首要解决的问题就是规划如何才能通过 Internet 把这些不同的应用程序集成起来，也就是要为这些程序提供通信协议可以理解的服务描述书。很多 IT 企业一直在致力于解决这一难题，从而形成了一个正快速增长的 Web 服务市场，这也是现代电子商务中增长得最快的业务。例如著名的 B2C 电子商务组织 Gartner 声称："采用 Web 服务将降低成本，将 IT 项目的效率提高 30%。估计 Web 服务业已形成了十多亿美金的市场规模，并正在迅速成长。"

所以，正加速开发新的协议和语言以标准化描述 Web 服务。目前，开发了一种基于

XML 的 SOAP(http://www.w3.org/TR/SOAP)协议为 Web 服务之间提供基于互联网的标准调用方法。此外,正在抓紧研发新的 Web 服务描述方法和 Web 服务架构语言,这也是 W3C(World Wide Web Consortium)组织当前的主要任务。

语义网将在广泛分发 Web 服务时提升 Web 服务的作用。许多 Web 服务供应商希望通过 Internet 在更大的范围内为不同的用户共享他们的服务,提供中间代理服务即一种让 Web 服务能在用户间自动匹配的能力,是非常困难的。而且,这种周而复始的同类词汇间的映射会导致数据库的暴露。使用现有的实现方法,Web 服务描述了输入、输出、端口和其他调用概要,但是服务的行为描述却以一个 content(内容)字段保留下来,等待着将来加以描述。因此,这个问题就和前面的数据库间整合非常相似了,不同企业用户间未经商定的不同映射等待着被解析。在预先分派好的团体用户里面还有达成一致的可能,但和那些外来的 Web 服务提供商,由于他们使用了不同的 schema,要统一建立映射关系就非常困难,这要求我们在整个供应链条上进行大量的预先约定,而这将大大限制 Web 服务的应用范围。

对此,强大的语义网表达语言能够提供帮助。RDF 的扩展,即 RDF schema 以及一种新研发的 Web 本体语言 OWL(http://www.w3.org/2001/sw/WebOnt/),能够建立层级和词库,帮助解释词汇间是如何关联的。例如,已在互联网上建立一个说明运送事件的 schema,邮寄是一种运输服务,加急邮件是一种邮政服务等等。通过合并不同的词汇表描述的服务可以轻松整合出新的服务,而且被合并的文件仍然是合法的 RDF。

此外,并不要求建立服务连接的描述信息采用自然语言中的公共词汇。不管外部服务来源是来自一个不同的用户还是开发者,是来自一个不同的词典还是随机在 Internet 上发现的某个 Web,都能解释映射信息。因此,只需将名为 lorry 的合作者和 truck 之间建立对应关系,以后当合并图表时,可以发现 lorry 与 truck 的联系。甚至,这种新的语言还允许执行更为复杂的映射和合并,例如如果把 Nissan-Maxima 定义为豪华型汽车,产地是日本,当我们连接到尼桑经销商的服务时,即可找到上面定义的属性。

当某些相对复杂的服务不能很快从 Internet 获得时,语义网将能提升现有 Web 服务的能力。例如一家专门提供小糖果礼品盒的公司需要同时订购 100 个心形巧克力和 200 个棒棒糖,并需要把它们运送到北京进行包装,很容易可以找到心形巧克力供应商、棒棒糖生产厂家甚至很多的运输企业,但这不是仅通过一个服务能够解决的。显然我们希望能把上述几个服务打包在一起而不必去辛苦寻找三个以上的 Web 服务。语义网允许把所需的服务轻松地整合起来,即便事先没有采用同样的词汇进行服务定义。甚至,语义网的应用程序还能分析实现目标的方法,提供高效、合理的 Web 服务集成(例如,送巧克力需要冷藏,能自动添加该项服务申请以保证巧克力不会在运输过程中融化)。尽管复杂的 Web 服务组合仍是一个尚在研究的课题,但许多基本的 Web 服务装配,如各种不同服务的输入和输出匹配已经可以通过现有的语义网工具成熟应用了。

或许,曾有人担心建立语义网是在从事一项面向未来且十分困难的工作,但事实并非如此。语义网,正如万维网一样,只要拥有明确的设想就可以在 Internet 上轻松实现。只是把很多众所周知的成熟技术带到 Internet 上,让不同的数据和应用程序通过 Web 自动集成,以消除以前需要复杂的人为干预才能协调工作的麻烦。

12.1.2 层次结构

语义网构建在自定义标记的 XML 和数据表示灵活的 RDF 基础上，旨在应用有效的标准和技术使计算机能够更多地理解 Web 信息，从而实现知识发现、数据集成和信息导航等，并将特定的信息添加到万维网上辅助服务的自动化。

语义网的层次结构如图 12.1 所示，自底向上依次为 Unicode（统一字符编码）和 URI（Universal Resource Indicator，统一资源定位符）、XML、RDF 和 RDF Schema（RDFS）、本体（Ontology）、逻辑（Logic）、证明（Proof）和信任（Trust）。在语义网的七层结构中，XML、RDF 和 Ontology 三层是核心和关键，主要用于表示 Web 信息的语义。经过长久的研究和发展，这三层已较为成熟，推出并形成了一系列的成果和标准，而证明层正处于探索之中。

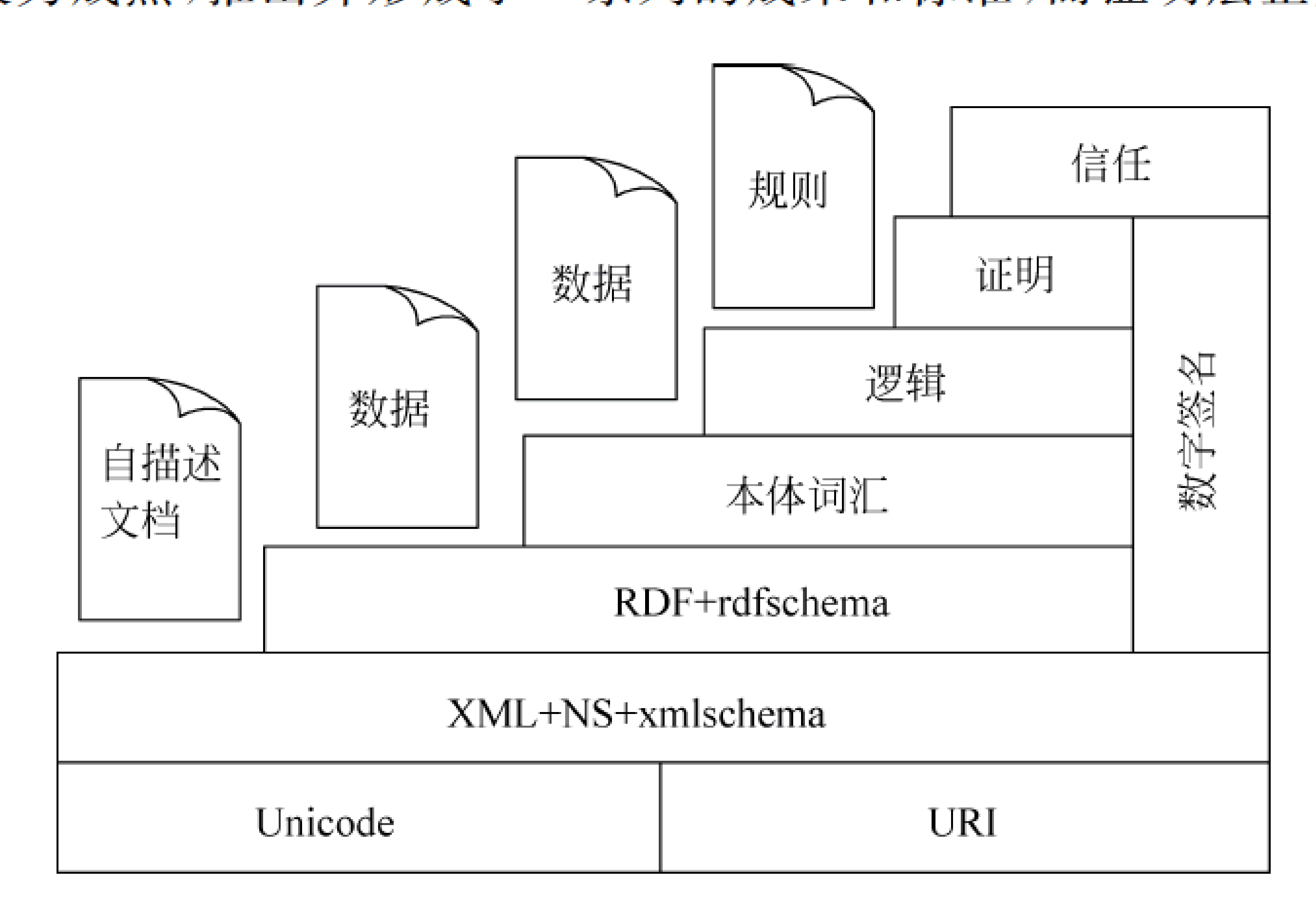


图 12.1 语义网的层次结构

此外，数字签名位于层次模型的右侧，贯穿了语义网的中间四层。数字签名是一种基于 Internet 的安全认证机制，用来检测文档是否被篡改，以证实其可靠性。当信息从一个层次传递到另一个层次时，可以使用数字签名说明信息的来源和安全性。这样，接收方可以通过数字签名鉴别其来源和安全性，决定信息的可信任程度。有了数字签名一些重要的电子商务活动就可在语义网上安全地进行。其实不仅对于语义网，数字签名对于所有的信息交换系统都非常重要。

1. Unicode 和 URI 层

Unicode 和 URI 是整个语义网的基础，其中 Unicode 负责资源的编码，URI 负责资源的标识。

Unicode 是一个字符集，该字符集中所有的字符都用两个字节表示，可以表示 65 356 个字符，基本上包括世界上所有语言的字符。语义网的最终目标是构建一个全球信息网络，必然涵盖各个国家和民族的语言，采用 Unicode 作为其字符编码方案，可从根本上解决跨地区、跨语言字符编码的格式标准问题。

URI 是语义网的根基。同 Internet 使用 URL（Universal Resource Locator）标识 HTML 页面一样，语义网同样需要一个类似的规范，唯一标识网络资源。语义网所使用的 URI 规范在 RFC 2396 中进行了详细说明。任何组织和个人都可以自由定义和使用 URI。

值得注意的是,URI 与 URL 区别很大。URL 用来标识网络路径,可以通过 Internet 在 URL 上访问到相应的资源;但一个 URI 标识的资源可能根本无法通过网络访问到,URI 标识仅仅标识一个资源,并不同时包含该资源的访问路径。URI 包含 URL,URL 是 URI 的超集,URI 支持语义网的对象和资源的精细标识,使精确信息检索成为可能。

2. XML、Name Space 和 XML Schema

XML 已经成为数据表示和数据交换的事实标准,提供一种格式自由的语法,用户可以按照自己的需要创建标记集,并使用这些标记编写 XML 文档。正因为任何人都可以自由定义标记,所以不可避免地会发生标记同名的情况。W3C 引入命名空间(Namespaces)机制,即在标记前面加上 URI 索引,以消解这种冲突。XML Schema 提供了一种对 XML 文档进行数据校验的机制,基于 XML 语法提供多种数据类型,对 XML 标记的结构和使用方法进行了规范。XML Query 是在 XML 基础上发展起来的技术标准,类似的还有 XPath 等,使用这些技术可以对 XML 文档进行数据检索、提取结点等操作。

然而,随着 XML 在数据交换和应用集成等领域的广泛应用,人们逐渐发现,XML 仅适用于表示数据的语法,却不能涵盖数据的语义。鉴于 XML 受到业界的普遍支持,并且已经具备了较完备的技术标准,在语义网的体系框架中将其作为数据表示的语法层。

3. RDF 和 RDF Schema 层

XML 不适于表达数据的语义,因此数据的语义定义和互操作需要由更高一层完成。因此,W3C 组织开发了一种新的语言以描述 Internet 的资源及其之间的关系,即资源描述框架(Resource Description Framework,RDF)。RDF 采用三元组又称为陈述(Statement),表示互联网的资源、属性和值。三元组由主体(Subject),谓词(Predicate)和客体(Object)组成,分别对应陈述中的资源、属性和值。RDF 提供了一套标准的数据语义描述规范,但它还需要定义描述中使用的词汇。RDF Schema(RDFS)提供了一种面向计算机理解的词汇定义,提供了描述类和属性的能力。RDFS 在 RDF 的基础上引入类、类之间的关系、属性之间的关系以及属性的定义域与值域等。如同一部字典,计算机通过它可以理解数据的含义。RDFS 明显区别于对象模型的是属性独立于类,一个属性可以应用于多个类或实例。

RDF Syntax 构建了一套完整的语法以便于计算机自动分析和处理,包括三种常用的表示方法,即图形、N3 和 XML。其中,图形表示是对 RDF 模型的直接描述,可以直接明了地观察 RDF 数据及其关系;N3 是一种三元组的方式,通过枚举 RDF 模型中的每个陈述表述 RDF 模型,最易于使用,简明易懂;RDF/XML 将 RDF 以 XML 语法描述,将 XML 的解析和解释过程相结合。这样,RDF 在帮助解析器阅读 XML 的同时,可以获取 XML 表达的语义,并可以根据其关系进行推理,从而做出基于语义的判断。但是 RDF/XML 常常因为过于复杂而难以使用。

该层用于描述万维网的资源及其类型,为 Web 资源描述提供一种通用框架和实现数据集成的元数据方案。最底层的 URI 标识 Web 的对象,RDF 和 RDFS 层则可对 URI 标识的对象进行陈述。

4. 本体(Ontology)层

该层用于描述各种资源之间的联系,本体揭示了资源本身及资源之间更为复杂和丰富的语义,从而将信息的结构和内容分离,对信息进行完全形式化的描述,使 Web 信息具有计算机可理解的语义。因为本体定义了不同概念间的关系,所以本体层能够对字典(或词汇)

的演化提供支持。

作为语义网中最为核心的一层，本体层在 RDF 和 RDFS 进行基本的类/属性描述的基础上，更进一步地描述本体以及它们之间的关系。这一层具有专用的本体描述语言，如 SHOE(Simple HTML Ontology Language)、OIL(Ontology Inference Language)、DAML(DARPA Agent Markup Language)以及 DAML+OIL 等。RDF 也是一种简单的本体描述语言，但其描述能力较弱，需要扩展。OWL(Web Ontology Language)是 W3C 推荐的本体描述语言，其实现较多地参考了 DAML+OIL 的设计思想和经验。

5. 逻辑(Logic)层

该层用来产生规则，主要提供公理和推理规则，为智能推理提供基础。

近年来，随着研究的不断深入，描述逻辑(Description Logic，DL)作为一种较为成熟的知识表示方法引入，对于 OWL 规范的制定起到了一定的指导作用。最近，研究人员已经开始尝试在 OWL 加入规则形成 OWL 的规则语言(OWL Rules Language，ORL)，以更好地实现自动推理。

6. 证明(Proof)层

证明层执行逻辑层产生的规则，主要提供认证机制，并结合信任层的应用机制评判是否能够信赖给定的证明。

证明层使用逻辑层定义的推理规则进行逻辑推理，得出某种结论。对于语义网的用户而言，这一推理过程应该是建立在可靠的数据基础上，应该是公开的，而且推理得到的结论也应该是可验证的。

7. 信任(Trust)层

主要提供信任机制，以保证用户代理(Agent)在 Web 上进行个性化服务和交互更安全可靠。

在语义网内进行推理并最终得出的结论应该是可以信任的，这需要满足：

(1) 可以信任所见的数据，即上下文；

(2) 可以信任所做的推理过程。

满足上述两点，才可以信任最终得到的推理结果。使用语义网的 RDF 模型，任何人都可以对任何资源进行描述，不同立场的人对相同的资源可能会做出完全相反的描述。信任层负责为应用程序提供一种机制，以决定是否信任给出的论证。信任层的建立，使智能代理在网络上实现个性化服务，以及彼此间的自动交互具备了可靠性和安全性。

12.1.3 元数据

元数据是描述数据的数据，是对万维网信息的一种描述方式，是机器可理解的信息。

元数据由一系列属性或元素组成，以实现查询、阅读、交换和共享。例如图书馆元数据——图书馆目录，包括一系列描述书籍和书面的数据，如作者、出版日期、出版社和书名等元素。

元数据与其所描述的资源之间的关联方式主要有两种：

(1) 元素包含在独立于该资源的记录中，如图书馆目录。

(2) 数据嵌在资源本身。

Internet 中元数据的概念非常流行,其重要作用在于:

(1) 组织和管理网络信息,挖掘信息资源,通过元数据可以在万维网上准确地识别、定位和访问信息。

(2) 查询所需信息。

(3) 组织和维护一个机构对数据的投资。

(4) 建立数据目录和数据交换中心。通过数据目录和数据交换中心等提供的元数据,用户可以共享,维护及优化数据等。

(5) 提供数据转换的信息。用户在获取信息的同时可以得到元数据,通过元数据,可理解信息和自身信息集成在一起,进行科学分析和决策。

元数据的编写是有标准的,通常不同领域根据不同的需求制定一种或多种标准。标准的制定可实现数据的交换和共享。一些重要的元数据标准包括 MARC(Machine Readable Cataloging,机器可读编目)和 Dublin Core 等。

(1) Dublin Core

美国在线计算机图书中心(Online Computer Library Center,OCLC)从用户的角度出发,创建了一种新的网络资源描述标准或格式,都柏林核(Dublin Core,DC)元数据标准应运而生。

当前,DC 元数据集包括 15 个核心元素,分别是:

• TITLE——对象的名称,由创建者或出版商给出。
• SUBJECT——对象所涉及的主题,包括资源或对象的关键字。
• DESCRIPTION——资源内容的描述。
• SOURCE——对象的来源。
• LANGUAGE——文字内容采用的语言。
• RELATION——和其他对象的关系。
• COVERAGE——对象的空间位置和时间持续性特征。
• CREATOR——资源内容的责任人。
• PUBLISHER——能获取对象的责任代理。
• CONTRIBUTOR——主要负责对象文字内容的人。
• RIGHTS——资源权限管理的声明。
• DATE——发布日期。
• TYPE——对象的类型。
• FORMAT——对象的数据格式。
• IDENTIFIER——唯一标识对象的字符串或数字,如 URL 或 URI。

(2) HL7

HL7(Health Level 7)是由美国国家标准局授权的标准开发机构 HL7 研发的一个专门用于医疗卫生机构及医用仪器、设备数据传输的标准。

HL7 适用于医院内部不同医疗信息系统之间交换病例资料、临床检验结果和财务信息,便于医院内部信息的交换和管理。同时,HL7 也适用于医院与医院、医院与保险公司、医院与上级主管部门之间大量信息交换的需要。

HL7 可应用于多种操作系统和硬件环境,也可以进行多个应用系统之间文件和数据的

交换，所有不同平台的医院信息管理系统通过 HL7 都可以顺利交互。采用 HL7 作为标准的 HIS 和医用仪器、设备可以实现无缝连接和医学信息的无障碍交换。

HL7 的应用不仅使医院内部不同系统间的交互大大简化，更便于各医院以及医院与其他机构之间的联系。

（3）IMS

IMS 是一个全球性的学习组织，以发展及推广开放性规范（Open Specification）为主要任务，主要在发展和推广有关教育的开放规范，以促进在线分布式的学习活动。IMS 全球学习联盟有两大目标：其一是达成分布式学习环境下应用系统或服务的互操作性，定义、发展所需的技术规范；其二是协助其他机构将 IMS 规范纳入产品或服务中。

由于 IMS 并非开发性组织，所制定的规范需要由正式的标准制定机构（如 IEEE 等）进行公开、公正的讨论和审核，投票通过后才能成为正式的标准。目前，IMS 开发完成的规范分别是 IMS 学习资源元数据说明（IMS Learning Resource Metadata Specification）、IMS 企业说明（IMS Enterprise Specification）、IMS 学习者信息包装说明（IMS Learner Information Package Specification）和 IMS 问题与测试互操作说明（IMS Question&Interoperability Specification）。

实现元数据的技术手段是 XML 和 RDF。XML 从数据和文档的底层实现格式化，保证从处理到交换的一致性，有利于在网络环境下采用通用的搜索引擎等工具，为实现广义数字图书馆（虚拟数字图书馆）提供了可能。

由于不同领域甚至同一领域存在多个元数据标准，当在不同元数据标准描述的资源体系之间检索时，则存在元数据的互操作性问题。利用特定的转换程序对不同的元数据标准进行转换，称为元数据映射（Metadata Mapping/Crosswalking）。目前出现了大量的转换程序，实现各种元数据标准之间的转换。例如 DC 与 USMARC、DC 与 EAD（Encoded Archival Description，编码档案描述）、DC 与 GILS（Government Information Locator Service，政府信息定位服务）、GILS 与 MARC TEI、Header 与 MARC、FGDC（Federal Geographic Data Committee）与 MARC 等。也可以利用一种中间格式对多种元数据进行转换。相比之下，格式映射转换准确、效率较高。

元数据的类型有四种，即内容元数据、管理元数据、负载信息元数据和参考信息元数据，它们从不同的维度、不同的层次描述电子文档或资源。其中：

（1）内容元数据——描述对象内容的信息。

（2）管理元数据——描述与电子文档相关的信息。

（3）负载信息元数据——提供电子文档的物理属性。

（4）参考信息元数据——源自电子文档中的超链接。在此“链接”的概念扩展到更一般的概念，用来表示任何万维网信息、文档和资源的参考链接。

12.1.4 核心技术

语义网的体系结构中，XML 层、RDF 层以及本体层是最核心的，它们是语义网知识表示的基础，为上层的推理和验证等奠定了基础，也是目前发展较为成熟的三种技术，下面将分别简要介绍。

1．XML

传统的 Internet 以 HTML 格式存储和组织分布式的文档，这带来的主要问题是：

- 任何人只要发现 HTML 不足以满足其需求时，就简单地在文档中增加标签，结果导致大量、非标准的 HTML 的出现。
- HTML 的标签主要是面向显示的，并不包含语义，因此很难让机器抽取内容以及自动地处理文档。

为了解决上述问题，W3C 开发了 XML 标准，并确定 XML 为语义网底层的数据交换格式。XML 通过制定标准容许用户自行定义标签，并通过文档类型定义（Data Type Definition，DTD）或 XML Schema 约束这些标签的内部结构，并解决了不同应用之间命名冲突的问题。XML 成功地实现了文档的内容与表示的分离，成为应用之间交换数据的最佳选择。

XML 的可扩展性、自我描述性及良好的结构定义，为语义网提供了完美的底层数据交换格式，并通过 XML Schema 规定交换数据的数据结构。然而 XML 并不能对所使用的标签提供语义解释，对机器语义理解没有太大帮助。因此，基于 XML 构建了 RDF 标准，实现对信息资源的语义描述。

2．资源描述框架语言

Web 是计算机可读的，但不是计算机可理解的，因此 Web 信息难以实现计算机自动处理。解决这一难题的途径是采用元数据索引 Web 信息，然后使用资源描述框架（Resource Description Framework，RDF）描述元数据与元数据之间的关系。RDF 是处理元数据的基础，可在应用程序之间交换机器可理解的 Web 提供了可互操作性。RDF 的应用广泛，如采用 RDF 的智能主题可提高机构之间知识共享和交换的能力。

RDF 定义一个简单的数据模型，通过性质（Property）和值（Value）描述资源以及资源与资源之间的关系。如果将 RDF 的性质看作是资源的属性，则 RDF 也可以看作传统的＜属性，值＞模型。此外，RDF 还可以描述资源与资源之间的关系，因此 RDF 类似于一个实体关系图。

RDF 的表示方法有三种，即图示、模型和三元组。在 RDF 模型中，资源以资源标识符表示，资源标识符由一个唯一资源标识符和一个可选的锚（Anchor）ID 组成。

RDF 提供了一个开放的表达 Web 资源的元数据描述模型，是由一系列的陈述，即主体-谓词-客体三元组组成。它可以表达 Web 上可标识的任何资源，如标题、作者、Web 的版权和注册信息、语言、格式和条目等。RDF Schema 是一个描述 RDF 资源的属性和类的词汇表，提供了关于这些属性和类的层次结构的语义，从某种程度上拓展了这种资源描述的能力，可以看成是轻量级的 Web 本体语言。

RDF 和 XML 的主要区别在于：

(1) XML 不包含语义，RDF 包含语义。

判断是否包含语义，需要知道语义的含义，机器可理解的语义是指由符号表示的对象与对象之间的关系；而语法则是指符号与符号之间的关系。

之所以说 XML&XML Schema 不包含语义，是指它们并不能使机器理解对象与对象之间的含义；而 RDF&RDF Schema 之所以包含语义，是指它们能表达标签所对应的对象之间的含义，而且包含了谓词逻辑并支持推理，如图 12.2 所示。

```
<rdf:Description    rdf:about =
"http://www.famouswriters.org/twain/mark">
<s:hasName>Mark Twain</s:hasName>
<s:hasWritten rdf:resource = "http://www.books.org/ISBN0001047582">
</rdf:Description>
<rdf:Description rdf:about = "http://www.books.org/ ISBN0001047582">
<s:title>The Adventures of Tom Sawyer</s:title>
<rdf:type rdf:resource = "http://www.description.org/schema#Book">
</rdf:Description>
```

图 12.2　RDF 示例

如图 12.3 所示的例子表达的是 Mark Twain 写了 The Adventures of Tom Sawyer 一书，即在二者之间建立了“hasWritten”的关系，这种关系也可以在 RDFS 中进行描述，这种关系能在 RDFS 与其他词汇建立联系(如父子关系、定义域、值域等)。

上述的例子以 XML 表示，如图 12.3 所示。

```
<description>
<hasName>Mark Twain</hasName>
    <hasWritten>
        <description>
            <book>
                <title>The Adventures of Tom Sawyer</title>
            </book>
        </description>
    </hasWritten>
</description>
```

图 12.3　XML 示例

通过上述结构，人们可以明显地看出其中的关系，但机器仍然无法理解 Mark Twain 和 The Adventures of Tom Sawyer 是什么关系，而只知道它们分别是树结构中的第二级结点和第五级结点。

因此，RDF 用来描述资源(或者对象)，并建立它们之间的语义关系。

(2) XML 的结构是树，RDF 的结构是图。

RDF 的三元组在文件中出现的顺序是随意的，三元组有多个谓词和客体时，其在陈述中被定义的顺序也是随意的；而 XML 中一个结点出现的顺序却不能更换。

(3) XML Schema 定义的是 XML 的词汇表，而 RDF Schema 定义的是词汇类型。

XML Schema 定义了 XML 中的词汇及其在树状结构中的位置关系，而 RDF 的词汇集太大了，大到 RDF Schema 无法描述，因此 RDF Schema 定义的是 RDF 中的词汇类型，并且定义了概念间的语义，如概念的父子关系、定义域及作用域等。

然而，同其他知识表示语言相比 RDF Schema 显得过于简单，语义表达能力不够，需要更上层对其语义解释能力的进一步扩展。

3. 本体

本体是支持知识共享和重用的形式化结构，将信息的结构和内容分离，实现对信息进行完全形式化的描述，为信息提供一个统一的共同表达的语义结构。

本体的主要作用是:

- 为人类和应用系统提供一个对于主题的共同理解。
- 为不同来源的信息合成,提供一个共同的相关领域的理解。
- 为不同的应用程序之间共享信息和知识,描述应用程序的领域,定义术语及其关系。

对于以概念的共享和理解为核心的语义网,本体提供了语义知识的明确化表示方法,因此本体在语义网中处于核心支配地位,在12.2节中将详细介绍本体。

12.1.5 开发工具——Jena

Jena是一套开发语义网应用的Java API,包括对RDF、RDFS和OWL描述的本体模型的解析、创建和串行化等,SPARQL语句的解析,转化为SQL以及基于规则的推理引擎。

Jena提供了读取、创建和输出模型的Java API。其主要的数据结构是图,但是用户操作主要还是在模型上进行。基本的方法是通过ModelFactory建立模型,然后通过模型创建资源,再通过资源添加属性,逐步构成一个图。

通过model.listStatement()方法可以得到所有Statement的游标,可以遍历模型中所有的三元组。但在取出三元组各部分时需注意,对象可以是字符也可以是资源,需要测试。

RDF图中的一个结点或者本体中的一个资源,在RDF中以<rdf:Description rdf:about =${uri}>开始,uri指出了资源的URI。如果下面有匿名资源,或者说图中有空白结点则用<${prediction} rdf:nodeid="${anonymousenodeid}">,另外再使用rdf:Description描述该空结点,即<rdf:Description rdf:nodeid="${anonymousenodeid}">。当然,人工编写时通过嵌套可以不用创建匿名资源。

12.1.6 Web 3.0

Internet经历了翻天覆地的重大变革。伴随着从Web 1.0向Web 2.0的过渡,Web 3.0已开始逐渐步入人们的视野。Web 2.0虽然只是互联网发展阶段的过渡产物,但正是由于Web 2.0,让人们可以更多地参与到Internet,特别是在内容上的创造。在这一点上,Web 2.0是具有革命性意义的。正是因为更多的人参与到了有价值的创造活动,那么要求互联网价值的重新分配将是一种趋势,因而必然催生新一代Internet——Web 3.0。

Web 3.0开发者们的目标是建造一个能针对简单问题给出合理、完全答复的系统。Web 3.0标准的核心是:

- 继承Web 2.0的所有特性。如以用户为中心,用户创造内容,广泛采用AJAX技术,广泛采用RSS内容聚合,表现为BLOG大行其道,Internet上涌现大量的个人原创日志。
- 帮助用户实现其劳动价值。目前的Web 2.0几乎都是用户免费劳动,免费生产内容娱人娱己。用户很难通过Web 2.0网站把自己辛辛苦苦生产的内容兑换成真实货币。Web 3.0的首要任务是让他们不再浪费劳动力,实现劳动价值。
- 网站无边界,遵守Web 3.0标准的网站可以方便地在数据、功能上实现彼此的互通、互动。未来的Internet是合作、共赢、资源互补、互促的Internet。分久必合,有相关

利益的网站会联合起来，趋于一体化。一个强有力的、方便的对外交互的标准是每个 Web 3.0 网站都必须实现的。

- 具备更清晰、可行的盈利模式。现在的 Web 2.0 网站大部分没有清晰可行的盈利模式，这是商业网站的致命弱点。有些 Web 2.0 网站有一些广告收入，但是规模不够，注定会被淘汰。
- 文字不仅限于 Internet 应用，这是 Web 3.0 标准的外延，可以应用到其他非互联网行业。

Web 3.0 的三个前提是：

- 博客技术为代表，围绕网民互动及个性化体验的互联网应用技术的完善和发展。
- 虚拟货币的普及，以及虚拟货币的兑换成为现实。
- 大家对网络财富的认同，以及网络财务安全的解决方案。

Web 3.0 与 Web 2.0 一样，不是技术的创新而是观念的创新，进而引领技术的发展和应用。Web 3.0 将催生新的王国，不再以地域进行划分，而是以兴趣、语言、主题、职业、专业进行聚集和管理的王国，可谓是“轮流坐庄”，每个用户都有机会打造出一个新的互联网王国而成为一个国王，也有可能在互联网王国的民主竞选中成为“总统”，到时将拥有来自全球各个角落的网络公民。

过去的 Web 1.0 采用超链接解决了信息孤岛的连接问题，Web 2.0 解决了网络发言权的解放问题，Web 3.0 则是要解决海量信息在细化后的定向搜索与获利机制问题。当 Web 3.0 为用户提供了更好的提升自我的整合能力后，这意味着能够更好地成为一个围绕用户服务的整合中心，这正是我们期待未来的 Web 3.0 受到用户欢迎的原因。

Web 1.0 被 Web 2.0 重新洗牌后，人人都有话语权，但是谁来听、谁来买单才是根本。也许正是因为还不涉及利益分配的原因，Web 2.0 得以迅速发展。同样，因为 Web 2.0 还没有形成良性的商业回报机制，所以一个能解决“利益分配”问题——将主要利益分配给最有贡献的内容提供者的 Web 3.0 便应运而生。

随着 Internet 的日新月异，Web 3.0 将是彻底改变人类生活的互联网形式。Web 3.0 使所有网民不再受到现有资源的限制，具有更加平等地获得财富和声誉的机会。事实上，已经有了 Web 3.0，只不过是了解不多，如电子商务和在线游戏，不管是 B2C 还是 C2C 模式，网民利用 Internet 提供的平台进行交易，整个过程中他们通过 Internet 付出了劳动并收获了财富。在线游戏通过积分的方式，角色扮演者通过攻城掠寨、花费大量的时间不断修炼，他们在那里可以获得声誉和财富，而这个财富通过一定的方式可以在现实中兑换，正所谓人生如同一场游戏，Internet 会让人们的生活变得更像游戏一样。当前的论坛也提供积分，但由于缺乏个性，不会成为未来的主流，最有代表性的博客，却在积分方面做得很少，劳动价值没有得到体现。为此，好的博主将另起炉灶，以便得到更多，这是在追求一种更加均衡的分配方式。

Web 3.0 究竟能够做些什么？DWS Group 推出其第一个 Web 3.0 的应用智能相册(Smart Albums)，它是一款简单易用的图片管理软件，集相册分类、图片标签和图片搜索等功能为一体，并可以日历模式展现，还可以对图片进行加密保存等。Web 3.0 在医疗领域已有应用，据调查美国的所有医学测试中，有 40%是因为不知道病人以前的测试结果而进行的。所幸的是，在电子病例领域，现在出现了 WorldVista 的开源标准，这个由 VA 开发的基

于 Vista 的标准向所有人开放，能够形成一个全球性的可互换医疗信息系统。Google 目前已完成了从信息制作、组织、存储、检索、发布、翻译和服务，以及无线服务 Google SMS 的一整条 Web 3.0 产业链的构建。Web 3.0 时代网络连接速度将达到 10Gbps，越来越多的家庭都用上了数据传输速率达 2～3Gbps 的连接，用户可以观看电影片花等多媒体内容，为 eBay、Salesforce.com 等互联网巨头开拓了新的市场。可见，Web 3.0 作为一种新的理念，正逐步融入我们的生活。

很多 Web 3.0 的尝试已经开始，但这些尝试聚合起来还需要一个漫长的过程，现在的 Web 3.0 如同一堆碎石，等待被整理为一条畅通的公路。我们今天描述 Web 3.0，就如同当年 Internet 诞生之前想象 Web 1.0 一样。

无论如何，我们已经奔驰在信息高速公路上，大家需要重新认识信息的本质。现在要做的是按照信息在现实中存在的属性和信息之间的关系结构建立公用信息标准，并按照这个标准搭建公用信息平台，通过不断地完善和升级，最终实现 Internet 的真正价值。

12.2 本体

本体是近年信息科学界最热门的词汇之一。在各种信息交互和集成、知识表示与获取的应用中，这一词汇频频出现，本体到底是什么？

12.2.1 哲学本源

本体的概念最初起源于哲学领域，并在很长一段时期都是哲学研究的一个分支。古希腊哲学家亚里士多德定义本体为“对世界客观存在物的系统的描述，即存在论”，即本体是客观存在的一个系统的解释或说明，所关心的是客观现实的抽象本质。为研究客观世界的存在问题，亚卫士多德、莱布尼茨、康德、皮尔斯和怀特海德等哲学家广泛地讨论了如何运用本体对现实世界进行分类、如何描述其中的物理实体、如何定义客观世界的抽象以及空间与时间的关系等问题。虽然，历史上对本体的思考主要是从哲学和逻辑学的角度进行，但前人的研究成果蕴涵了很多如何组织现实世界知识的方法，这为本体被信息科学所借鉴奠定了基础。逻辑在信息科学中占有重要地位，但人们认识到逻辑没有描述具体现实世界的能力，逻辑中的存在量词符号仅仅能声明某物存在，但逻辑本身却没有词汇来描述到底存在的是什么，本体由此被引入信息科学中，以弥补逻辑表达能力的不足。本体包含了观察与推理两个范畴。观察提供现实世界的知识，描述知识的组织形式；推理通过虚拟框架澄清观察的意义，描述知识的语义。

20 世纪 90 年代，信息科学的发展面临着种种新难题，诸如知识表示、知识共享和复用等。特别地，由于 Internet 的飞速发展，如何组织、管理和维护海量信息并为用户提供有效的服务成为一项重要而迫切的研究课题。本体作为一种能在语义和知识层次上描述信息系统的概念模型的建模工具，引起了国内外众多研究者的关注，并在计算机领域得到了广泛应用，如知识工程、数字图书馆、软件复用、信息检索和 Web 异构信息的处理和语义网等。

12.2.2 定义

Ontology 这一哲学范畴，被人工智能赋予了新的定义，并引入到信息科学。然而信息科学界对 Ontology 的理解也是逐步发展并走向成熟的。1991 年 Neches 等人最早给出 Ontology 在信息科学中的定义是给出构成相关领域词汇的基本术语和关系，以及利用这些术语和关系构成规定这些词汇外延规则的定义。后来随着研究的深入，在信息系统、知识系统等领域对 Ontology 给出了不同的定义，如 1993 年 Gruber 定义 Ontology 为"概念模型的明确的规范说明"；1997 年 Borst 进一步完善为"共享概念模型的形式化规范说明"。Studer 等人对上述定义进行了深入研究，认为 Ontology 是共享概念模型的明确的形式化规范说明。

Studer 等人的本体定义包含四层含义：概念模型(Conceptualization)、明确(Explicit)、形式化(Formal)和共享(Share)。其中，概念模型是指通过抽象出客观世界中一些现象的相关概念得到的模型，其表示的含义独立于具体的环境状态；明确是指所使用的概念及使用这些概念的约束都具有明确的定义；形式化是指 Ontology 是计算机可读的，也是计算机可处理的；共享是指 Ontology 中体现的是共同认可的知识，反映的是相关领域中公认的概念集，针对的是团体而非个体。Ontology 的目标是捕获相关领域的知识，提供对该领域知识的共同理解，确定该领域内共同认可的词汇，并从不同层次的形式化模式上给出这些词汇(术语)和词汇之间相互关系的明确定义。尽管定义有很多不同的方式，但就内涵而言，不同研究者对于 Ontology 的认识是统一的，都将其看作领域(领域的范围可以是特定应用，也可以是更广的范围)内部不同主体(人、机器、软件系统等)之间进行交流(对话、互操作、共享等)的一种语义基础，即由 Ontology 提供一种共识，而且 Ontology 提供的这种共识更主要的是为机器服务，机器并不能像人类一样理解自然语言表达的语义，目前的计算机也只能把文本看成字符串进行处理。

到底什么是 Ontology？学术界对此并没有达成共识，引用较多的是 Gruber 的定义，即 Ontology 是一个共享的概念化规范，而概念化是指某个领域中的概念及其相互关系，是我们希望描述的世界的一个抽象的、简化的视图。

Ontology 与通常所谓的词典的区别在于：

- 词典是由人阅读的，Ontology 必须方便机器阅读。
- 词典注重概念本身的描述，Ontology 既注重概念本身的描述，又注重概念间之间关系的表示。
- 词典通常以自然语言描述，Ontology 可以用其他符号语言描述。

经过多年的努力，研究人员已经构建了一些有影响力的 Ontology。

12.2.3 建模

本体是一种组织知识的艺术。为研究如何利用本体组织知识，Perez 等人采用了分类法，并归纳出五个基本建模元语：类(Classes)或概念(Concept)、关系(Relation)、函数(Function)、公理(Axiom)和实例(Instance)。类或概念表示对象的集合；关系表示领域中

概念之间的交互作用；函数是一类特殊的关系，该关系的前 $n-1$ 个元素可以唯一决定第 n 个元素；公理代表永真断言；实例代表元素，就语义而言表示的就是对象。关系在本体中非常重要，从语义的角度，基本的关系有四种：part-of、kind-of、instance-of 和 attribute-of。part-of 表示概念之间部分与整体的关系；kind-of 表示概念之间的继承关系，类似于面向对象中父子类之间的关系；instance-of 表示概念的实例与概念之间的关系，类似于面向对象中对象和类之间的关系；attribute-of 表示某个概念是另一个概念的属性。实际建模中，概念之间的关系不限于上述的四种基本关系，可以根据领域的具体情况定义相应的关系。Ontology 正是通过这些建模元语组织现实世界的知识。

12.2.4 分类

目前，广泛使用的本体包括 WordNet、FrameNet、GUM 和 SENSUS 等。WordNet 是基于心理语言规则的英文词典，以在特定的上下文环境中可互换的同义词的集合为单位组织信息；FrameNet，英文词典采用称为 Frame Semantics 的描述框架，提供强大的语义分析能力，目前发展为 FrameNet II；GUM 面向自然语言处理，支持多语种处理，包括基本概念及独立于各种具体语言的概念组织方式；SENSUS 面向自然语言处理，为机器翻译提供概念结构，包括 7 万多个概念。

对于 Ontology 的分类则有着不同的标准。常用的本体划分准则是详细程度和领域依赖程度。详细程度是为了描述或刻画建模对象的程度，高的称作参考本体(Reference Ontology)，低的称作共享本体(Share Ontology)。根据依赖程度可以划分为四类：

- 顶级(Top-level)本体描述的是最普遍的概念及概念之间的关系，如空间、时间、事件和行为等，与具体应用无关，其他本体均为其特例。
- 领域本体(Domain Ontology)描述的是特定领域中的概念和概念之间的关系。
- 任务本体(Task Ontology)描述的是特定任务或行为中的概念及概念之间的关系。
- 应用本体(Application Ontology)描述的是依赖于特定领域和任务的概念和概念之间的关系。

12.2.5 构建方法

如何构建 Ontology？Gruber 提出了五条准则，即：

(1) 清晰性(Clarity)。本体必须有效地说明所定义术语的含义，定义应该是客观的，与背景独立的，当定义可以用逻辑公理表达时，应该是形式化的，定义应该尽可能地完整，所有定义应该用自然语言说明。

(2) 一致性(Coherence)。本体应该是一致的，换言之，应该支持与其定义相一致的推理，它所定义的公理以及用自然语言进行说明的文档都应该具有一致性。

(3) 可扩展性(Extendibility)。本体应该为可预料到的任务提供概念基础，应该支持在已有概念的基础上定义新的术语，以满足特殊需求，而无须修改已有的概念定义。

(4) 编码偏好程度最小(Minimal encoding bias)。概念的描述不应该依赖于某一种特殊的符号层表示方法，因为实际系统可能采用不同的知识表示方法。

(5) 约定最小(Minimal ontological commitment)。本体约定应该最小,只要能够满足特定的知识共享需求即可,这可以通过定义约束最弱的公理以及只定义通信所需的词汇来保证。

对于本体构建方法,大多数研究者都倾向于采用一种近似软件工程的方法。本体构建一般分为若干步骤,是一个不断迭代、逐步精炼的过程,主要步骤包括:

(1) 确定 Ontology 的目的和使用范围。

(2) Ontology 捕获:即确定关键的概念和关系,给出精确定义,并确定其他相关的术语。

(3) Ontology 编码:选择合适的语言表达概念和术语。

(4) 已有 Ontology 的集成:尽可能重用和修改已有 Ontology。

(5) 评估:根据需求描述、能力询问(Competency Question)等对 Ontology 以及软件环境、相关文档进行评价。

由于本体到目前为止仍处于相对不成熟的阶段,每一工程都有自己独立的方法。最常用的构建方法主要是骨架法、IDEF-5 法和循环获取三种,已得到业界的普遍认可,下面将分别介绍。

1. 骨架法

Mike Ushold 和 Micheal Gruninger 提出的骨架法(Skeletal Methodology)在企业本体基础上,是相关商业企业间术语和定义的集合。该方法只提供开发本体的指导方针。在构建过程中虽没有提出特有的评价方法,但是认为评价方法应该是其中的一个环节,具体步骤如下:

(1) 确定目的和范围(Identify Purpose and Scope)

在此阶段需要确定建立本体的目的、本体应用的范围以及用户群等。

(2) 建立本体(Building the Ontology)

这一阶段包括本体获取、本体编码和现有本体的集成。

(3) 评价(Evaluation)

没有提出自己的评价方法,只是认为评价应该是整个方法论的一个环节。

(4) 文档化(Documentation)

包括本体中定义的主要概念、元本体等。目前很多知识库和本体缺少文档也是一种知识共享的障碍,某些编辑器可以自动生成这些文档。

2. IDEF-5 方法

IDEF(ICAM Definition Method)的概念是在结构化分析方法的基础上发展而来的,1981 年美国空军公布的 ICAM(Integrated Computer Aided Manufacturing)工程首次使用了名为 IDEF 的方法。到目前为止,已经发展成为一个系列。IDEF-5 通过两种语言(即图表语言和细化说明语言)获取某个领域的本体论,提供过程流图和对象状态转移网图(OSTN)这两种图表获取、管理和显示过程。

IDEF-5 提出的本体构建方法包括五种活动,分别是:

- 组织和范围(Organizing and Scoping)

确定本体建设项目的目标、观点和语境,并为组员分配角色。

• 数据收集(Data Collection)

收集本体建设需要的原始数据。

• 数据分析(Data Analysis)

分析数据,为抽取本体做准备。

• 初始化的本体建立(Initial Ontology Development)

从收集的数据中建立一个初步的本体。

• 本体的精炼与确认(Ontology Refinement and Validation)

完成本体建设过程。

3. 循环获取

Alexander Maedche 等人提出的循环获取(Cyclic Acquisition Process)方法是一种环状结构,如图 12.4 所示。

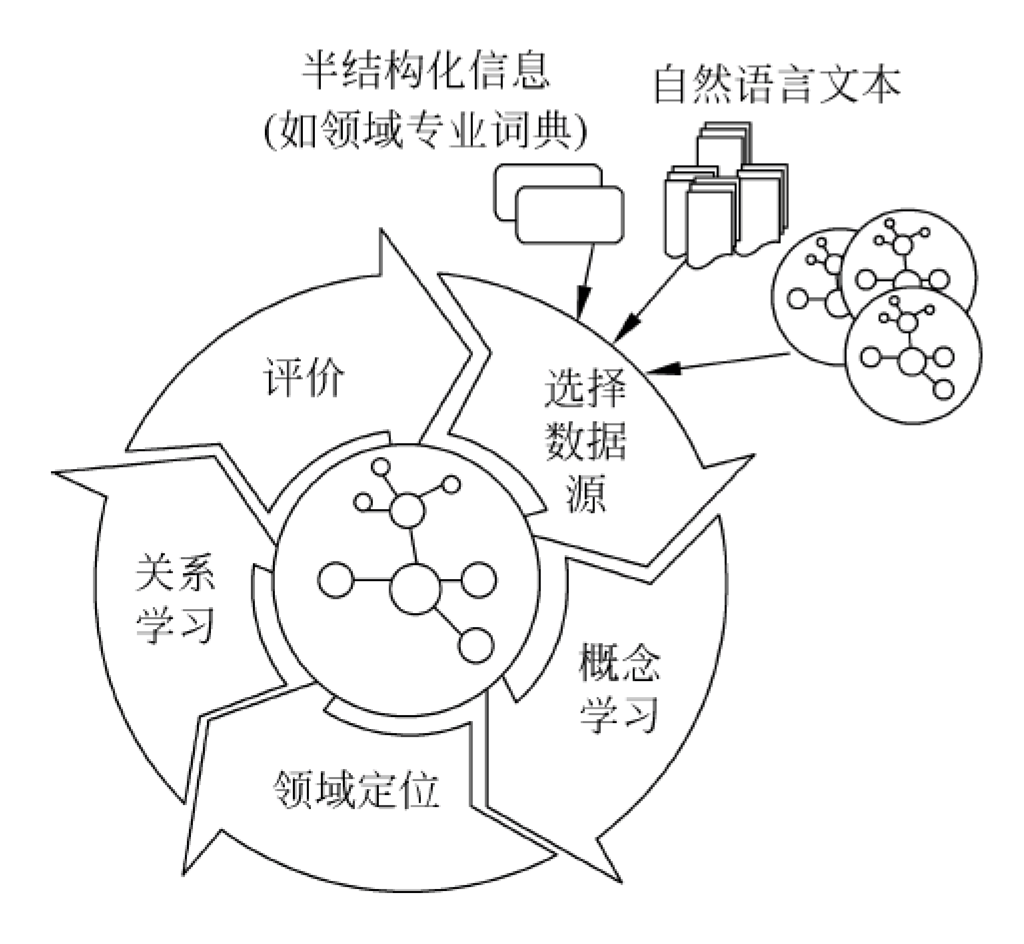


图 12.4 循环获取过程

具体过程如下:

(1) 环状的起点是一个通用的核心本体的选择。

任何大型的通用本体(如 Cyc、Dahlgren 的本体)、词汇语义网(如 WordNet、EuroWordNet、HowNet)或者领域相关的本体都可以作为这个过程的开始。选定基础本体后,用户必须确定用于抽取领域相关实体的文本。

(2) 从选择的文本中获取领域相关的概念,并建立概念之间的分类关系。

(3) 除去领域无关的概念,只留下和领域相关的。这时,建立了目标本体的概念结构。

(4) 从基础本体中继承一些关系,其他的关系需要通过学习的方法从文本中抽取。

(5) 对获得的领域相关的本体进行评价,还可以重复上述过程。

构建本体需要一种表示语言作为基本的形式化工具。原则上,本体能够用各种各样的语言实现,如非形式化的、半形式化的或形式化的,并无强制性的规范。目前,有两种逻辑常用作本体的形式化工具:其一是谓词演算及其变种;其二是框架逻辑。其中,谓词演算及其变种较为常见。如 M. R. Genesereth 和 R. E. Fikes 提出的 KIF(Knowledge Interchange Format)是一种基于扩展的谓词演算的形式语言,包含类、子类、属性、值、关系和公理等原

语；Ontolingua 本体就是使用 Frame Ontology，其语法基于 LISP；CYCL 是 CYC 的知识表示语言，类似于一阶谓词演算；XOL（XML-Based Ontology Exchange Language）是一种本体交换语言，提供本体定义格式的交换，基于 XML 语法；OIL（Ontology Inference Language）是基于框架的表示语言，目前还在不断完善中。

12.2.6 描述语言

下面以 OWL（Web Ontology Language，网络本体语言）为例介绍本体描述语言的功能特性。

OWL 是 W3C 推荐的语义互联网中本体描述语言的标准。W3C 总结了之前的 DAML＋OIL、RDF 和 RDFS 等几种语言的开发经验，在 2004 年 2 月正式推出 OWL。OWL 是语义网发展过程的一个重要里程碑，已经经过广泛的讨论并得到比较一致的认可。OWL 既保持了对 DAML-ONT/OIL/RDFS 的兼容性，又保证了更强大的语义表达能力，还保证了描述逻辑的可判定推理等。W3C 提出的本体语言栈如图 12.5 所示。

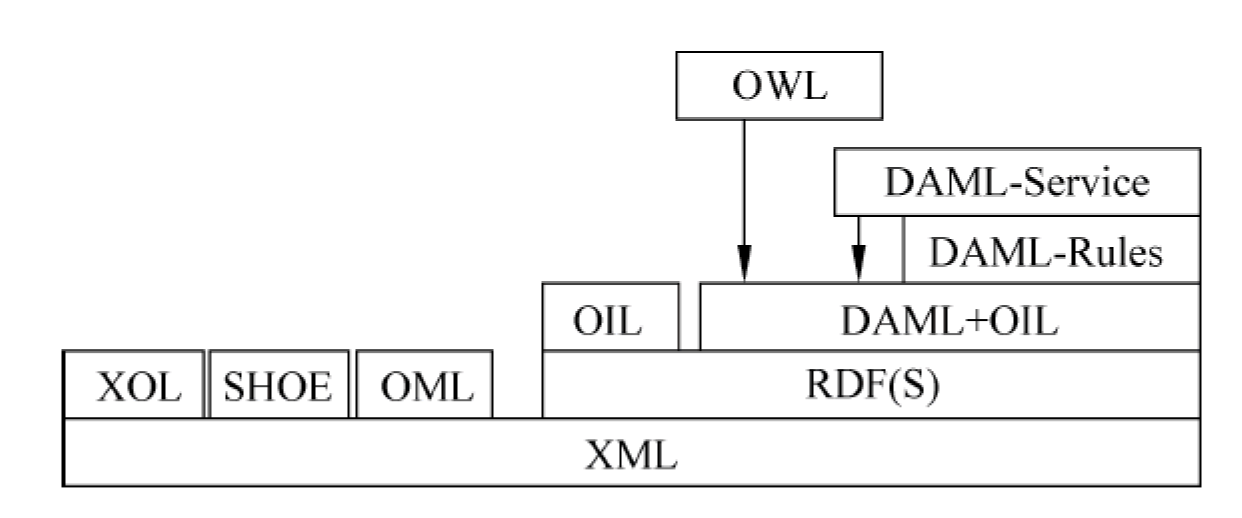


图 12.5　W3C 提出的本体语言栈

RDFS 提出类的概念，定义类和性质，可以描述其他类和性质，然而其表达能力非常有限，RDFS 局限于子类分层和属性分层，以及属性的定义域和值域限定，难以提供推理方面的支持。W3C 确定的语义网用例所需的表达能力要比 RDF 和 RDFS 强得多，OWL 由美国和欧洲等研究机构联合提出的 DAML＋OIL 语言发展而来，具有良好定义的语法，高率的推理支持，以及充分灵活的语义表达能力，从根本上解决了 RDF 和 RDFS 表达语义的局限性，成为 W3C 推荐的描述本体语言的标准。

OWL 和 RDF/RDFS 中一些建模原语的关系如图 12.6 所示。

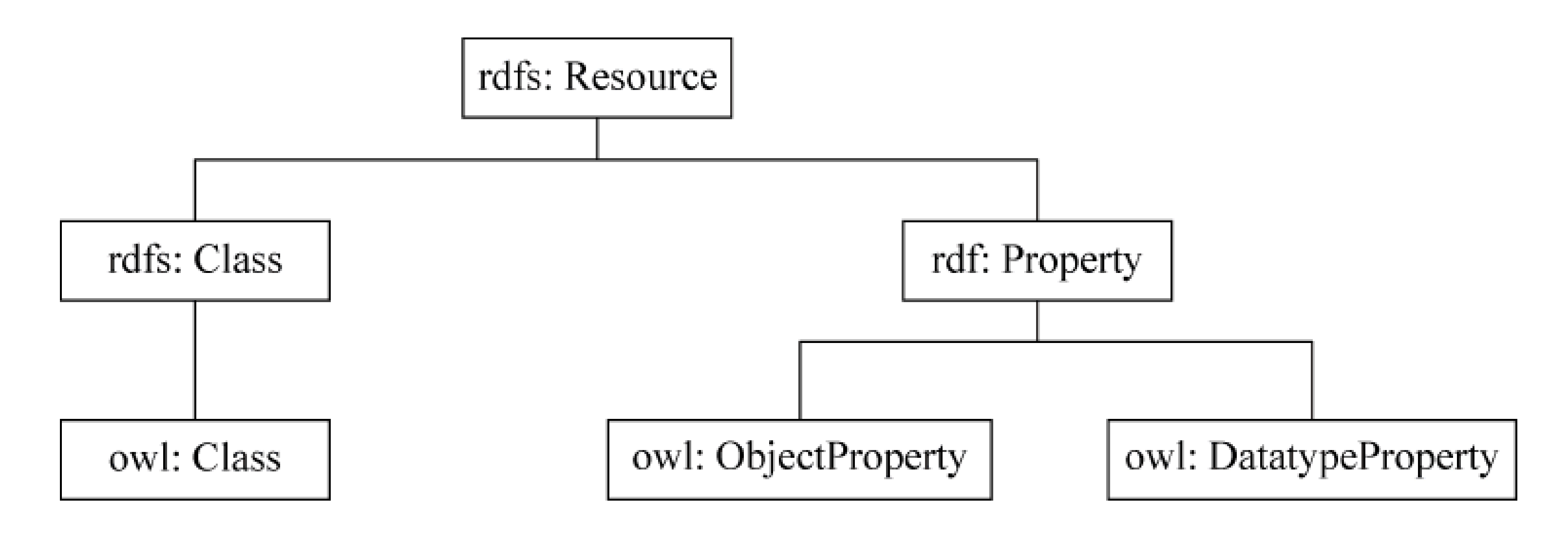


图 12.6　OWL 和 RDF/RDFS 建模原语的关系

RDFS 的局限性如下：

- 属性的局部辖域。rdfs:range 为一个属性（例如吃）定义的值域是相对于所有类的，无法定义只适用于某些类的值域限制（例如无法定义牛只吃植物，而其他动物还可

以吃肉)。

- 类不相交性。有时需要表示类的不相交性,例如男性和女性不相交。但 RDFS 只能规定类之间的子类关系,如女性是人类的子类。
- 类的布尔组合。有时希望通过对已有类的并、交或补等操作,组合产生新的类,例如定义人类为男性和女性的不相交的并,RDFS 则不容许这样的定义。
- 基数约束。有时需要对一个属性不同取值的个数加以约束。如一个人恰好有一个父亲和母亲,一门课程至少有一个授课者讲授,RDFS 同样不能表达这样的约束。
- 属性的特殊性质。有时需要规定属性具有传递性(如大于),唯一性(如母亲),或定义属性的逆属性(如吃和被吃)。

OWL 针对 RDFS 的局限进行了如下扩展:

- OWL 中使用 owl:import 可以导入其他本体,该语句具有传递性。
- 类元素——owl:Class 进行类定义; owl:disjointWith 表示类的不相交; owl:equivalentClass 表示类相等。此外,还包含两个预定义类 owl:Thing 是所有类的父类; owl:Nothing 是空类,是所有类的子类。
- 属性元素——有两种属性,即 owl:ObjectProperty 和 owl:DatatypeProperty。前者将对象关联起来,常表示为动作;后者将对象和属性值关联起来,owl 没有预定义的数据类型,使用 XMLS 的数据类型。可以定义属性的逆属性和等价属性,分别使用 owl:inverseOf 和 owl:equivalentProperty。
- 属性约束——通过 owl:Restriction、owl:allValuesFrom/owl:hasValue、owl:someValuesFrom 规定 owl:onProperty 指定属性。分别表示该 property 的取值必须"全部/至少有一个"来自某范围。owl:allValuesFrom 和 owl:hasValue 的区别在于前者指定的是类或者一个数据范围,而后者指定的是一个个体或者一个值。
- 类的交、并、补——owl:intersectionOf、owl:unionOf、owl:complementOf。
- 类的枚举——owl:one of。
- 类的实例——同 RDF。
- 基数约束——owl:Restriction、owl:minCardinality、owl:maxCardinality。
- 特殊性质——传递性 owl:TransitiveProperty; 对称性 owl:SymmetricProperty; 函数性 owl:FunctionalProperty(如规定同一个对象的该属性不可以取同一个值,如年龄、身高等); 逆函数性 owl:InverseFunctionalProperty,即规定不同对象的该属性不可以取相同的值,如身份证号。

OWL 和其他知识表示语言表达能力的比较如表 12.1 所示。

表 12.1 OWL 和其他知识表示语言表达能力的比较

比较项目	XML DTD	XML Schema	RDF Schema	OIL	DAML+OIL	OWL
有界列表				√	√	√
基数约束	√	√		√	√	√
类表达式			√	√	√	√
数据类型		√	√	√	√	√
已定义的类				√	√	√

续表

比较项目	XML DTD	XML Schema	RDF Schema	OIL	DAML+OIL	OWL
枚举	√	√		√	√	√
等价				√	√	√
可扩展性			√	√	√	√
形式化语义			√	√	√	√
继承			√	√	√	√
推理				√	√	√
本地化约束				√	√	√
条件约束					√	√
实例化			√		√	√

12.2.7 实例

目前，一些具有代表性的本体包括 WordNet、CYC、Sensus 和 HowNet 等。其中：

WordNet 是一个基于语言心理学原理的英语词汇数据库，可用作词典、推理词典和分类词汇数据库。

CYC 是世界上最大、最完善的常识知识库。

Sensus 是一个主要用于机器翻译的本体实例。

值得一提的是董振东先生积多年的心血构建的 HowNet(知网)是目前国内一个不可多得的共享本体。下面主要从基本概念、知识结构和概念关系三个方面介绍 HowNet。

1999 年年初，中国中文信息学会常务理事董振东先生在 Internet 上公布了自己的研究成果——HowNet。HowNet 是描述概念与概念之间的关系，以及概念的属性与属性之间的关系的知识库，支持中、英文两种语言，具有独特的知识表示方法。近些年，HowNet 的出现使人们对汉语语义的研究又盛行起来。

HowNet 把客观世界看作是由很多概念构成的。概念与概念之间有各种各样的关系，这些关系相互交织构成了一个网。这种以网状的结构组织概念的方式，使得简单的概念描述可以表达丰富的概念关系，这是 HowNet 区别于其他语义词典的本质特征。

HowNet 是一个常识知识库，用 KDML(Knowledge Dictionary Mark-up Language)语言作为其知识表示的语言，然而由于这种描述知识的方法并没有得到广大应用的支持，且对于 HowNet 作为知识库本身，没能够提供一套完好的添加知识实例的方法。因此，以 HowNet 原始系统作为知识库，距离实际的应用还有一段差距，但作为一种语言学资源，HowNet 可以发挥它的积极作用。所以我们更愿意把 HowNet 理解为一种定义完好的半结构化语义词典进行利用。

HowNet 有几个重要的概念，如概念、属性、义原和动态角色等。其中，预定义了一千多个义原，用来描述概念的静态特征，而动态角色用来激活概念的静态特征，使概念与概念之间以及概念间的属性之间动态联系。其中：

(1) 概念

即词语的意义,而词语是概念的形式。同一个概念可能有不同的词语形式,包括不同语言的。如“医生”和“doctor”表达的是同一个概念。没有意义的字也就没有概念,如“葡”。HowNet 中有 22 574 个概念,由中、英两种语言的各 70 000 多词语表示,总记录数是 150 100。虽然没有一个记录是重复的,但两种语言对应的词语是重复的。HowNet 中概念用一个英语和汉语词语的组合来确定概念的唯一性。

(2) 属性

即概念的特征。一个实体所具有的属性是多元的,正是属性的多元化体现了关系的多元化。例如“纸”有颜色这一属性,正是这一属性造成它可能与“笔”、“写”和“画”发生关系。在日常生活中人们还会用“纸”来点火,这是由它另外一个属性“易燃性”造成的,这时颜色的属性变得无关了。

(3) 义原

又称为义素,是意义的最小单位。HowNet 中所有概念定义的基本成分是义原。义原分为五类,每个类别的义原形成一个树状的层次结构。HowNet 中,概念的主要特征即第一个属性,由位于主要特征文件中的义原描述,如实体表、事件表、属性值表和属性表等文件中的义原。因此,HowNet 中概念的上下位关系,由其第一义原的层次结构体现,而概念的其他属性可由任意义原描述,包括次要特征义原、数量表和数量值表等。

(4) 动态角色

用于描述概念的动态特征,如图 12.7 所示的医生是医治的施事者,病人是医治的受事者。动态角色常用于标注复杂的概念,格式为“动态角色=义原”,其中的动态角色可以为“agent/施事者”、“patient/受事者”、“instrument/工具”、“LocationFin/终处所”等。

不同于 WordNet、EuroWordNet 等国外盛行已久的语义词典,HowNet 除了能很好地支持中文,另一个特点是面向计算机的网状的知识系统,这是它与其他树状的词汇词典的本质不同。HowNet 的网状知识体系如图 12.7 所示。

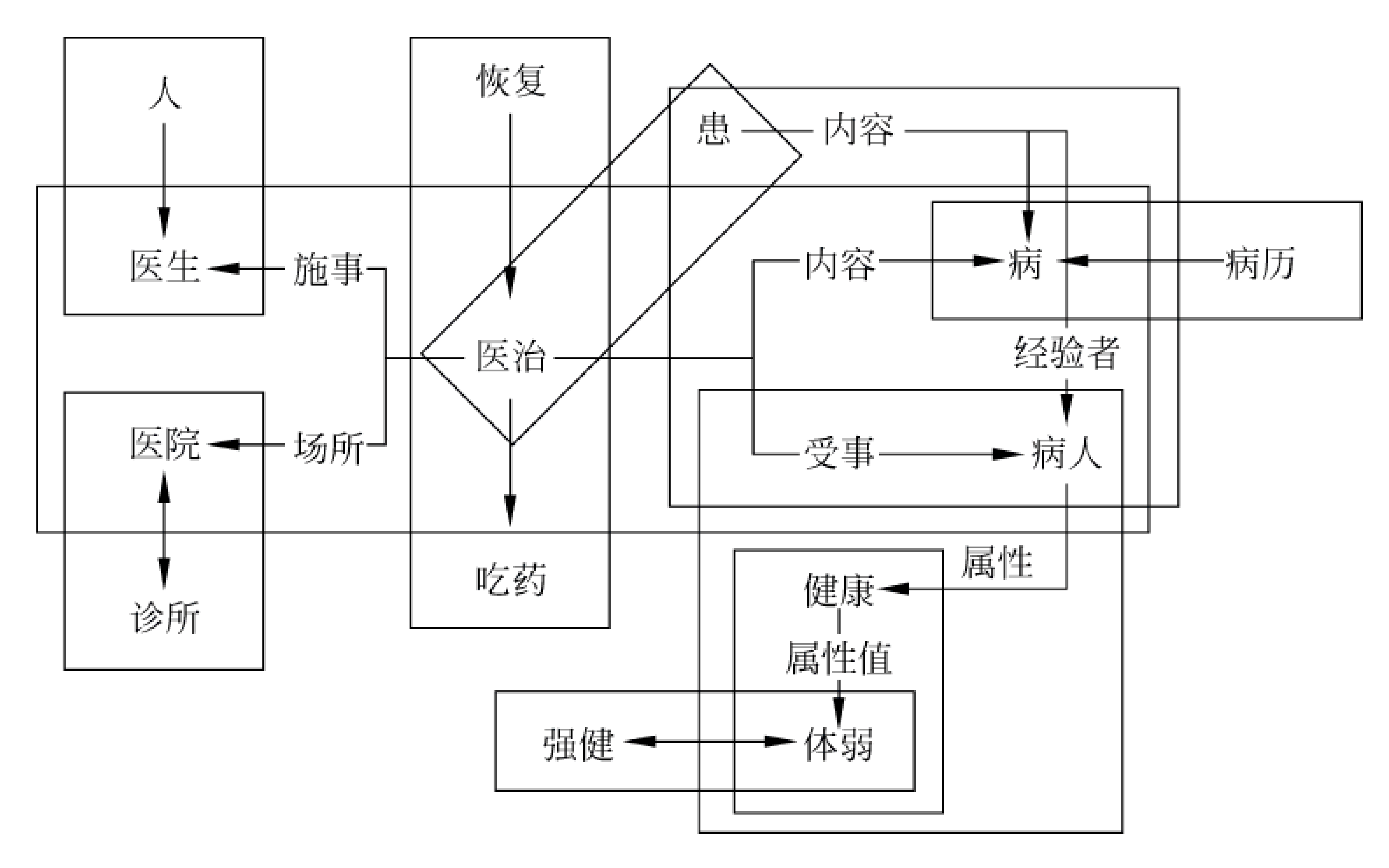


图 12.7 HowNet 的网状知识体系结构

如图 12.7 所示的网状知识结构在 KDML 中的表达方式如图 12.8 所示。

```
医生:
DEF = {human|人:domain = {medical|医},HostOf = {Occupation|职位},{doctor| 医治:agent = {～}}}
患者:
DEF = {human|人:domain = {medical|医}, {SufferFrom|罹患:experiencer = {～}}, {doctor|医治:patient =
{～}}}
医院:
DEF = {InstitutePlace|场所:{doctor|医治: location = {～},content = {disease|疾病}}, domain = {medical|
医}}
病历:
DEF = {document|文书:{record|记录:content = {disease|疾病},LocationFin = {～}}, domain = {medical|医}}
健康:
DEF = {Health|健康: host = {AnimalHuman|动物}}
多病:
DEF = {unhealthy|不健}
病:
DEF = {disease|疾病}
药:
DEF = {medicine|药物}
```

图 12.8　HowNet 概念的定义

其中,每个概念的定义项都由义原和动态角色描述。如 content、LocationFin、domain 等为动态角色,而出现在定义项中由双语标注的是义原。在 HowNet 中,义原通过一个树状的层次结构构成上下位关系,也具有同概念一样的定义格式,如图 12.9 所示。

```
| {HealthValue|健康值}
| ---- {healthy|康健}
| --------- {unhealthy|不健}

disease|疾病:
DEF = {phenomena|现象: {doctor|医治:content = {～}} ,
     {SufferFrom|罹患: content = {～}},
     RelateTo = {medicine|药物}{Health|健康}{HealthValue|健康值},
     domain = {medical|医}}
medicine|药物:
DEF = {artifact|人工物:{doctor|医治:instrument = {～}},RelateTo = {disease|疾病},
     domain = {medical|医}{chemistry|化学}}
```

图 12.9　HowNet 的义原层次结构和定义

通过这样的定义方式,使概念的关系图容易被计算机所理解,让计算机通过理解概念的语义作为其提供智能服务的基础。

概念关系是 HowNet 的灵魂,正是因为有了丰富的概念关系,HowNet 才包含丰富的语义。HowNet 描述概念之间多种类型的关系,有显性的关系,也有隐性的关系,还有组合的关系。

显性关系通常通过动态角色体现如下:

• 部件-整体关系,通过动态角色 PartOf 描述。如同火车有部件-整体关系的词语包括

餐车、车厢、豪华车厢、客车、车厢、旅客车厢、寝车、卧车、火车头、蒸汽机车和机车。

- 属性-宿主关系,通过动态角色 HostOf 描述。
- 材料-成品关系,通过动态角色 MaterialOf 描述。
- 施事-事件关系,通过动态角色 Agent 描述。
- 受事-事件关系,通过动态角色 Patient 描述。
- 工具-事件关系,通过动态角色 Instrument 描述。
- 场所-事件关系,通过动态角色 Location、LocationIni、LocationFin 等描述。
- 时间-事件关系,通过动态角色 Time、TimeAfter、TimeBefore 等描述。
- 值-属性关系,无须动态角色,直接标注。

HowNet 定义了百余个动态角色与特征表示概念的动态关系。

HowNet 的许多关系隐含在特征文件和概念词典描述中,通过一些特殊的结构和位置信息体现出来,如概念间的上下位关系通过定义项中第一个义原的层次结构体现出来,而对义关系则通过对义关系中的两两特征对体现出来。

HowNet 的隐性关系包括:

- 上下位关系——即概念间的父子关系,如“苹果”是“水果”的下位概念,“水果”是“植物”的下位概念。
- 同义关系——指的是不同的词语形式表达同一种概念。如“西红柿”和“番茄”是同义词,而“西红柿”和“红薯”不是同义词,只是同类词。HowNet 中,通过中、英文词形以及定义项体现同义关系,如表 12.2 所示。

表 12.2　HowNet 概念定义的实例

Id	中文词形	英文词形	定　　义
089505	西红柿	tomato	{ part|部件: PartPosition={embryo|胚}, whole={vegetable|蔬菜}, {eat|吃:patient={～}}}
025550	番茄	tomato	{ part|部件: PartPosition={embryo|胚}, whole={vegetable|蔬菜}, {eat|吃:patient={～}}}
036732	红薯	yam	{ part|部件: PartPosition={embryo|胚}, whole={vegetable|蔬菜}, {eat|吃:patient={～}}}

- 对义关系——HowNet 把反义之间没有灰度的称之为对义,一般由事件体现,如是/非、买/卖、教/学等。对义关系是根据 HowNet 的对义词组提取。
- 反义关系——HowNet 把反义之间有灰度的称为反义,一般由属性值体现,如大/小、美/丑等等。反义关系是根据 HowNet 的反义词组提取。

除了上述关系外,HowNet 还可以表达一些复杂的组合关系。所谓的复杂概念是以事件为中心,除了事件中心本身以外还有一个或一个以上的动态角色。

在对 HowNet 的研究中,只需从我们能利用到的角度提取其中一部分,将其转换为对应的 OWL 形式,使其从一个半结构化的语义词典转换为一个高层本体。如果需要,还可以对原始文件进行更深层次的研究,并提取出很多其他有价值的关系。

例如，采用 Protégé 本体编辑工具，Protégé 是斯坦福大学开发的本体编辑与知识获取工具，带有 OWL 插件的 Protégé 可以支持 OWL 格式的本体编辑与输出，并通过 Racer 工具进行本体的一致性检查和推理。将 HowNet 的概念、义原、动态角色以及词性，分别映射到 OWL 的类、对象属性以及数据类型属性，并通过类的交、并、约束等性质完成复杂概念的定义。

下面说明这些术语在 Protégé 以及 OWL 语言中是如何表示的。

(1) 义原的表示

如图 12.10 所示，将义原的实体类、事件类、属性类、属性值类以及第二特征类的层次结构表示出来。

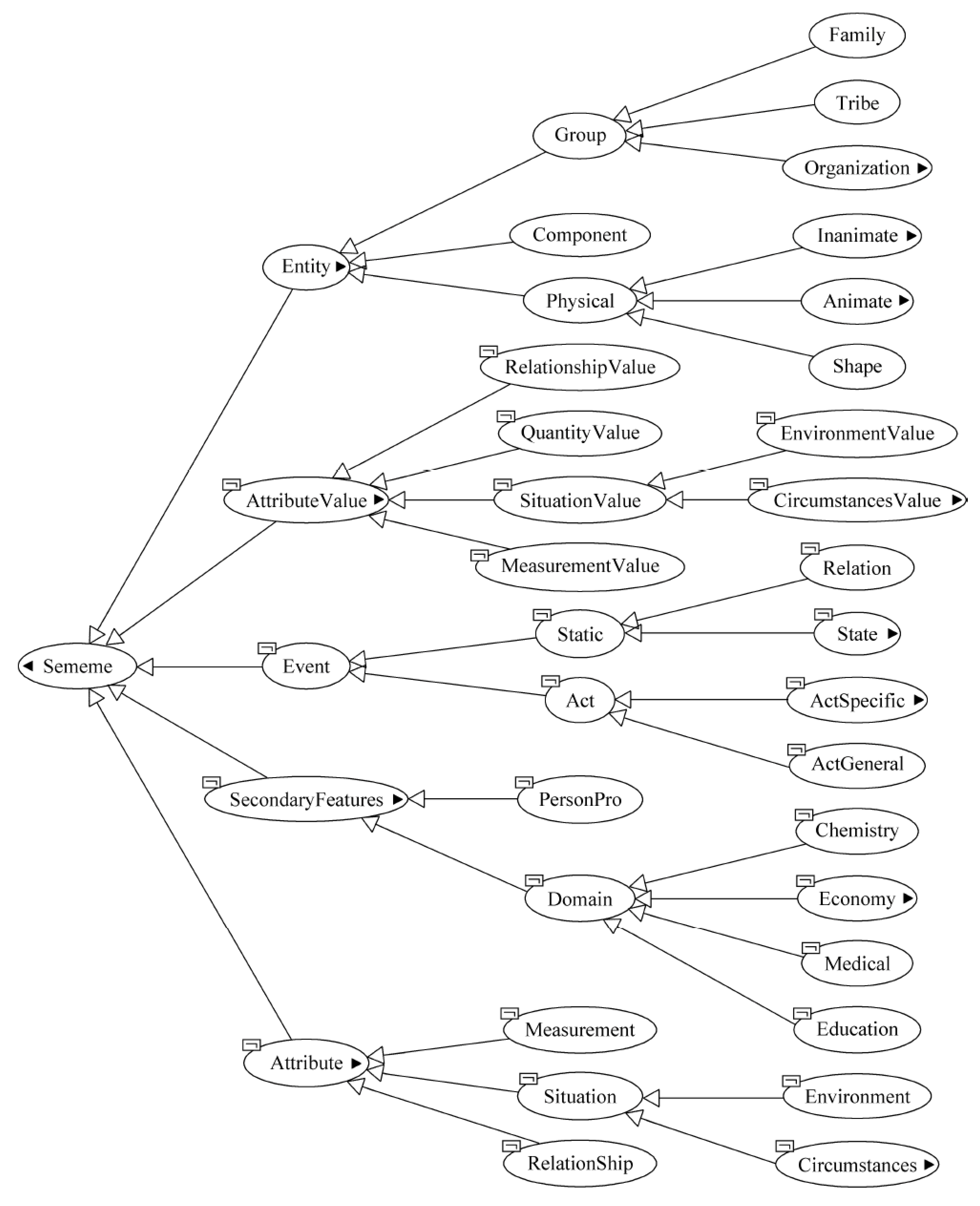


图 12.10　HowNet 义原的在 Protégé 中的表示

(2) 概念的表示

以“医院”为例,医院的定义项为 DEF＝{InstitutePlace|场所:{doctor|医治:location＝{～},content＝{disease|疾病}},domain＝{medical|医}},其在 Protégé 中的表示如图 12.11 所示,对应的 OWL 表示如图 12.12 所示。

InstitutePlace
Word
DOMAIN only Medical
inverse_of_LOCATION only (Doctor and (CONTENT only Disease))

图 12.11　HowNet 中的概念“医院”在 Protégé 中的表示

```
<owl:Class rdf:ID = "hospital">
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource = "#inverse_of_LOCATION"/>
                <owl:allValuesFrom>
                    <owl:Class>
                        <owl:intersectionOf rdf:parseType = "Collection">
                            <owl:Restriction>
                                <owl:onProperty rdf:resource = "#CONTENT"/>
                                <owl:allValuesFrom rdf:resource = "#Disease"/>
                            </owl:Restriction>
                            <owl:Class rdf:about = "#Doctor"/>
                        </owl:intersectionOf>
                    </owl:Class>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource = "#DOMAIN"/>
                <owl:allValuesFrom rdf:resource = "#Medical"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf rdf:resource = "#InstitutePlace"/>
        <rdfs:subClassOf rdf:resource = "#Word"/>
        <rdfs:comment rdf:datatype = "&xsd;string">医院</rdfs:comment>
</owl:Class>
```

图 12.12　HowNet 中的概念“医院”的 OWL 表示

对于概念的表示,还有一种较为简化的方法,即对于类似于 location＝{～}的定义项中带有“～”的概念,表示动态角色对应的是本身,可以通过将该概念添加到对应的义原的定义中,如可将“hospital”的“inverse_of_LOCATION”属性“doctor|医治”转化为“doctor|医治”的“LOCATION”属性,因为“inverse_of_LOCATION”同“LOCATION”具有相对的意义。

(3) 动态角色的表示

HowNet 中动态角色的表示如图 12.13 所示。若应用前面提及的简化的概念表示方法,可以不必设置动态角色的逆属性。

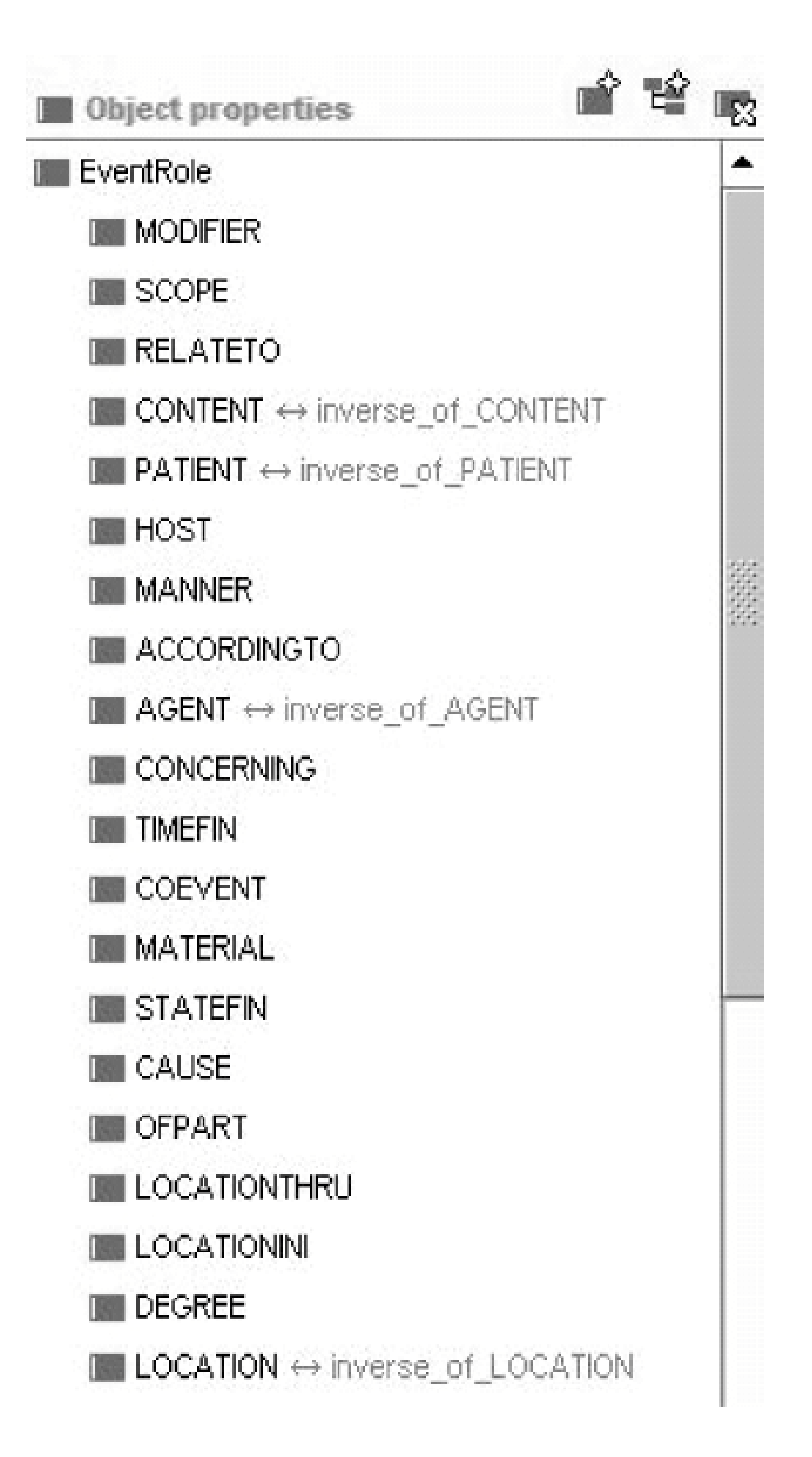


图 12.13　HowNet 动态角色定义为 owl：ObjectProperty

（4）词性以及语言的表示

如图 12.14 所示。

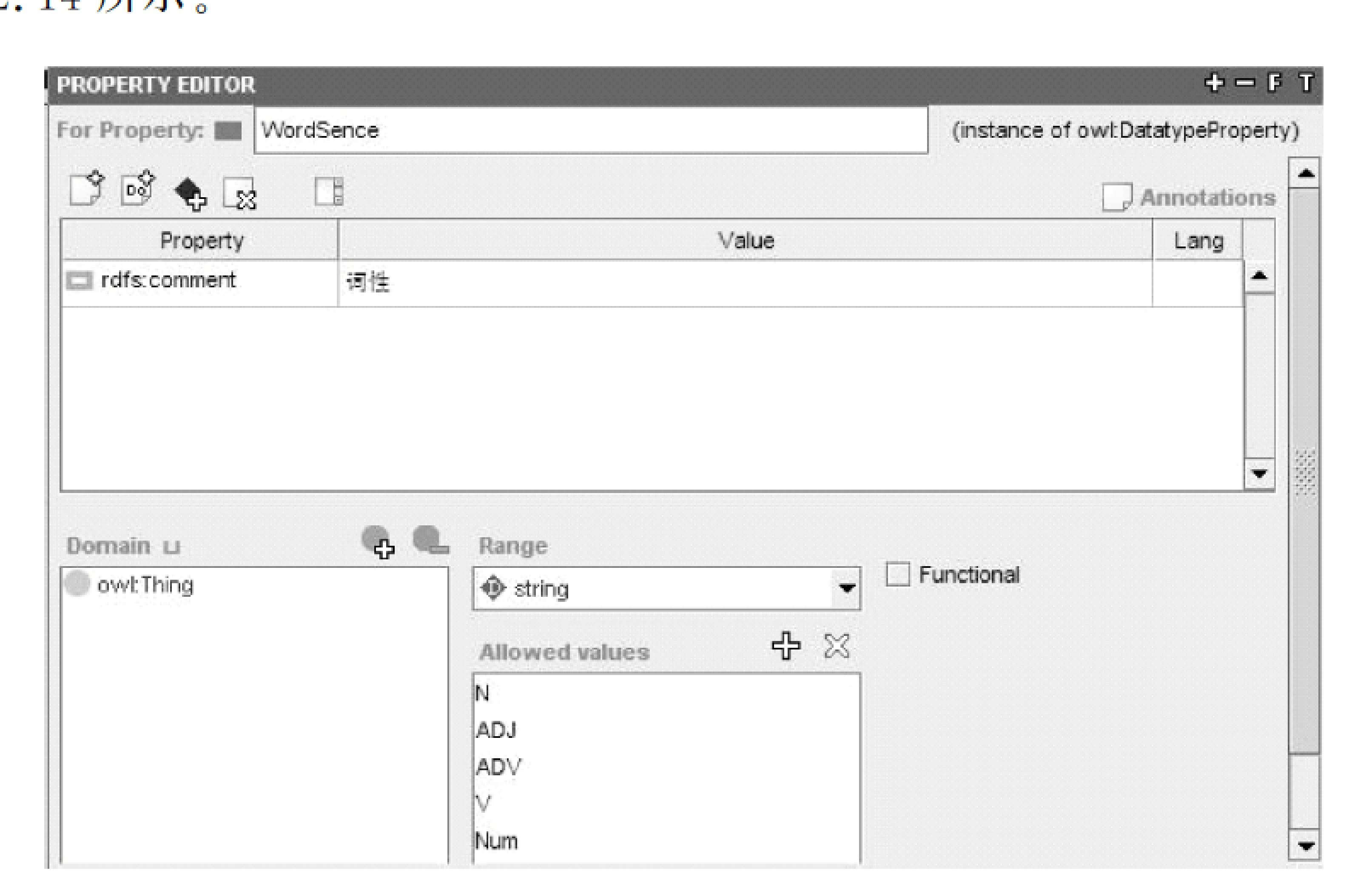


图 12.14　HowNet 词性表示为 owl：DatatypeProperty

参 考 文 献

[1] 陈莉,焦李成. Interne/Web 数据挖掘研究现状及最新进展. 西安电子科技大学学报(自然科学版),2001,28(1):114-118

[2] 池太崴. 数据仓库结构设计与实施. 北京:电子工业出版社,2005

[3] Craig S. Mullins. 数据库管理——实践与过程. 北京:电子工业出版社,2003

[4] D. Dou,D. V. McDermott,and P. Qi. Ontology Translation on the Semantic Web. Journal of Data Semantics,2:35-57,2005

[5] Dimitrios Skoutas. Alkis Simitsis. Designing ETL Processes Using Semantic Web Technologies. DOLAP'06,November 10,2006,Arlington,Virginia,USA,pp. 67-74

[6] Dong X,Halevey A. Data integration with uncertainty. In:Proc. of the 33rd Int'l conf. on Very Large Data Bases (VLDB 2007). New York:ACM Press,2007,687-698

[7] Dong Zhendong,Qiang Dong,HowNet,http://www. keenage. com

[8] E. R. Hruschka and N. F. F. Ebecken. Applying a Clustering Genetic Algorithm for Rules from Supervised Neural Network. Proceedings of the IEEE International joint Conference on Neural Networks (IJCNN'2000). Como,Italy,July 2000

[9] 飞思科技产品研发中心. Oracle9i 数据仓库构建技术. 北京:电子工业出版社,2003

[10] 黄曾阳. 概念层次网络理论. 北京:清华大学出版社,1998

[11] J. Han,M. Kamber. Data Mining:Concepts and Techniques. Beijing:Higher Education Press,Morgan Kaufmann Publishers,2001. 25-40

[12] John Poole Dan Chang. 公共仓库元模型数据仓库集成标准导论. 北京:机械工业出版社,2004

[13] 林杰斌. 数据挖掘与 OLAP 理论与实务. 北京:清华大学出版社,2003

[14] 鲁川. 现代汉语的语义网络. htp://www. hnc. com

[15] 陆汝钤. 人工智能. 北京:科学出版社,1996

[16] MartinT. Hagan,Howard B. Demuth,Mark H. Beale. 神经网络设计. 北京:机械工业出版社,2002

[17] McGuinness DL,van Harmelen F. OWL Web ontology language overview. World Wide Web Consortium (W3C) Recommendation. 2004. http://www. w3. org/TR/owl-features/

[18] Philip A. Bernstein,Laura M. Haas. Information Integration in the Enterprise. Communications of the ACM,2008,Vol. 51,No. 9,pp. 72-79

[19] Ralph Kimball,Margy Ross 等著. 数据仓库工具箱:维度建模的完全指南(第二版). 北京:电子工业出版社,2003

[20] Ralph Kimball,Laura Reeves,Margy Ross. 数据仓库生命周期工具箱:设计、开发和部署数据仓库的专家方法. 北京:电子工业出版社,2004

[21] 史忠植. 知识发现. 北京:清华大学出版社,2002

[22] 宋炜等. 语义网简明教程. 北京:高等教育出版社,2004

[23] W. H. Inmon. Building the Data Warehouse (Fourth Edition). Wiley,2005

[24] 袁毓林. 语言的认知研究和计算分析. 北京:北京大学出版社,1998

［25］ Zheng Yan，Huang Ronghuai，Zhan Xiaosu，Zhang Yan. A Clustering Algorithm for Customer Behavior Analysis. The 6th International Conference on Computer-Aided Industrial Design and Conceptual Design（CAID&CD'2005），pp. 1791-1794

［26］ Zheng Yan，Cheng Xiaochun，Huang Ronghuai and Man Yi. A Comparative Study on Text Mining Methods. The 2nd International Conference on Advanced Data Mining and Applications，August 14-16 2006，Xi'An，China. pp. 644-65